НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

«На правах рукопису»
УДК 004.043

«До захисту допущено»

Завідувач кафедри
_____ І.Р. Пархомей
(підпис)
“___”_____2019 р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 121 «Інженерія програмного забезпечення»

на тему: Інтелектуальна система розміщення товарних позицій на складі

Виконав: студент другого курсу, групи ІТ-84мп
(шифр групи)

Говоровський Сергій Валентинович _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник доцент, к.т.н., доцент Корнага Я.І. _____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____ _____
(назва розділу) (науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

Рівень вищої освіти – другий (магістерський)

Спеціальність 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ І.Р. Пархомей
(підпис)
«___»_____2019 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Говоровському Сергію Валентиновичу
(прізвище, ім'я, по батькові)

1. Тема дисертації «Інтелектуальна система розміщення товарних позицій на складі» _____
науковий керівник дисертації__доцент, к.т.н., доцент Корнага Я. І.,_____
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «___»_____ 2019 р. №_____

2. Термін подання студентом дисертації _____

3. Об'єкт дослідження – процес роботи складу.

4. Предмет дослідження – алгоритм оптимального розміщення товарних позицій на складі.

5. Перелік завдань, які потрібно розробити – аналіз проблеми та існуючих рішень; аналіз і реалізація методу; розробка методу; розробка програмного забезпечення; дослідження ефективності розробленого методу.

6. Орієнтовний перелік ілюстративного матеріалу – шість плакатів

7. Орієнтовний перелік публікацій – одна публікація

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

9. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Аналіз предметної області	13.09.2019 р.	
2	Постановка задачі	15.09.2019 р.	
3	Аналіз інформаційного забезпечення	20.09.2019 р.	
5	Аналіз алгоритмічного забезпечення	25.09.2019 р.	
6	Розробка алгоритмічного забезпечення	15.10.2019 р.	
7	Розробка програмного забезпечення	01.11.2019 р.	
8	Маркетинговий аналіз стартап-проекту	10.11.2019 р.	
9	Висновки	15.11.2019 р.	

Студент _____ С. В. Говоровський
(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ Я. І. Корнага
(підпис) (ініціали, прізвище)

АНОТАЦІЯ

У роботі розглянуто проблему області систем керування складськими приміщеннями, показано основні особливості існуючих рішень та додатків, їх переваги та недоліки.

Для інтелектуального розміщення нових товарних позицій на складі розроблена нейронна мережа з можливістю в автоматичному режимі знаходити нове місце на складі для товару. Це дозволяє зекономити час та ефективно використовувати корисну площу складського приміщення.

Визначено завдання для системи інтелектуального розміщення товарних позицій на складі за допомогою нейронної мережі та відібрано технології, за допомогою яких можна реалізувати дане завдання. Описано структуру програмного забезпечення.

Ключові слова: нейронна мережа, система управління складом, веб-додаток, база даних, машинне навчання.

Розмір пояснювальної записки – 86 аркушів, містить 8 ілюстрацій, 31 таблиці, 6 додатків.

ABSTRACT

Examines the problem of of the area of warehouse management systems, shows the main features of existing solutions and applications, their advantages and disadvantages.

For the intelligent placement of new product positions in the warehouse a neural network has been developed with the ability to automatically to find a new place in the warehouse for the items. This saves time and efficient use of the storage space.

The tasks for the system of intellectual placement of the item positions in the warehouse by means of a neural network are determined and the technologies by which it is possible to realize this task are selected. The structure of the software is described.

Keywords: neural network, warehouse management system, web application, database, machine learning.

Explanatory note size – 86 pages, contains 8 illustrations, 31 tables, 6 applications.

Пояснювальна записка
до магістерської дисертації

на тему:

Інтелектуальна система розміщення товарних позицій на складі

Київ – 2019 року

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

WMS (Warehouse Management System) – система управління складом;

MLP (Multi-layer Perceptrons) – нейромережа з багатошаровими перцептронами;

IT – інформаційні технології.

СУБД – система управління базами даних;

БД – база даних;

SQL (Structured Query Language) – мова структурованих запитів;

MVC (Model-View-Controller) – модель–представлення–контролер, шаблон проектування;

API (Application Programming Interface) – прикладний програмний інтерфейс;

HTML (Hypertext Markup Language) – мова розмітки гіпертекстових документів, стандартна мова розмітки для створення веб-сторінок;

CSS (Cascading Style Sheets) – каскадні таблиці стилів.

ВСТУП

Сьогодні ми не уявляємо жодного дня без покупок. Сфера споживання розвивається стрімкими темпами. Кожного дня мільйони людей купують, передають і доставляють мільйони товарів. З року в рік цифра обороту товарів тільки зростає. Разом з цим розвивається також сфера онлайн покупок, тобто зроблених в Інтернеті на спеціальних майданчиках, інтернет-магазинах. Ще декілька десятиліть тому люди навіть уявити не могли, що всього за декілька кліків можна отримати товари які забажаєш, або які є необхідними.

Відповідно, для того щоб від виробника продукція поступила до споживача вона має пройти декілька етапів. Зазвичай вони є наступними: доставка від виробника до реалізатора продукції; реалізація, тобто продаж товару; доставка від продавця до споживача. Ключовою ланкою в кожному з цих етапів шляху товару від моменту створення до отримання покупцем є місця, де цей товар якийсь час зберігається. Тобто мається на увазі склад. Без складування (в тому чи іншому вигляді) не може відбуватися жодний логістичний процес, оскільки об’єкти перевезення необхідно розміщувати та відвантажувати для подальшої доставки.

З основних проблем роботи складу можна зазначити облік продукції, яка наявна і зберігається в даний момент часу, а також оптимальне та правильне розміщення товарів, що тільки прибули для подальшого зберігання. Проблема правильного вибору локації для складських одиниць особливо гостро стоїть в умовах малої корисної площі складу, тобто при досить невеликому приміщенні.

Метою роботи є створення інтелектуальної системи визначення оптимального місця для одиниці товару на складі з урахуванням його затребуваності та попиту. Така система дозволить знаходити найбільш правильне місце для нових товарів, що дасть змогу ефективніше використовувати корисну площу та полегшить пошук цього товару по складі. У свою чергу це здешевить вартість зберігання та позитивно відіб’ється на кінцевій ціні продукції.

В даній роботі описуються етапи проектування та розробки системи для інтелектуального розміщення товарний позицій на складі з урахування затребуваності товару та його попиту.

Задля успішної реалізації системи необхідно вирішити наступні завдання:

- проектування структури бази даних;
- написання нейронної мережі для пошуку місця;
- навчання та тестування штучного інтелекту;
- написання функціональної частини застосування;
- розробка веб-інтерфейсу для користувачів.

Після проведення аналізу існуючих мов програмування та засобів розробки веб-додатків обрано технологію ASP.NET із фреймворком MVC та мовою програмування C#. В якості СУБД вирішено використовувати Microsoft SQL Server.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ

1.1 Об’єкт та предмет дослідження

В даній дипломній дисертації створюється спеціальна система для вирішення проблеми оптимального розміщення товарних позицій на складі з урахуванням попиту та затребуваності товару.

Склад – це місце, яке використовується для зберігання або накопичення товарів. Складування також може бути визначене як взяття на себе відповідальності за зберігання товарів. Зберігаючи товари протягом якогось часу та звільняючи їх за потребою, складування приносить велику користь у процесі логістики.

До основних функцій складу можна віднести наступні:

1. Зберігання. Це основна функція складування. Надлишки товарів, які не потрібні в даний момент часу, можуть зберігатися на складах. Вони можуть поставлятися за потреби [1].

2. Стабілізація цін. Склади відіграють важливу роль у процесі стабілізації цін. Це досягається створенням часової корисності шляхом складування. Це дозволяє уникнути падіння цін на товари, коли їхня пропозиція є в достатку, і зростання цін в нестійкий сезон.

3. Зменшення ризиків. При зберіганні товарів на складах вони піддаються багатьом ризикам у вигляді крадіжок, зносу, розвідки, пожежі тощо. Склади будуються таким чином, щоб мінімізувати ці ризики. За будь-які втрати або збитки, понесені товарами, власник складу несе відповідальність перед власником товару.

4. Класифікація та упаковка. Склади забезпечують пакування, переробку та сортування товарів.

Необхідно зберігати товари так, щоб вони були доступними для покупців коли це потрібно. Певна кількість товару зберігається на кожному етапі маркетингового процесу. Належна та адекватна організація роздрібної торгівлі товарами у ідеальному стані має важливе значення для успіху в маркетингу.

Зберігання дозволяє фірмі продовжувати виробництво в очікуванні попиту в майбутньому [2].

Склади дають можливість підприємцям продовжувати виробництво протягом року та продавати свою продукцію, коли є достатній попит. Потреба в складі виникає ще й тому, що деякі товари виробляються лише в певний сезон, але користуються попитом протягом року. Так само певна продукція виробляється протягом року, але затребувана лише протягом певного сезону. Складування сприяє виробництву та розповсюдженню у великих масштабах.

Щоб визначити складську логістику, ми повинні спочатку зрозуміти значення логістики. У найпростіших термінах логістика може бути визначена як детальне планування, організація, управління та здійснення складних операцій. У багатьох галузях промисловості, включаючи складування, логістика також поширюється на потік як фізичних товарів, так і інформації [3].

Таким чином складська логістика охоплює всі найрізноманітніші складні фактори – організацію, рухи та управління – залучені до складування. Сюди входить потік (доставка та отримання) фізичних запасів, а також поступлення більш абстрактних товарів, включаючи інформацію та час.

Логістика складу може також поширюватися на будь-що – від поводження з пошкодженими товарами, політики безпеки до управління людськими ресурсами. Іншими словами, складська логістика включає всі процедури та організаційні інструменти, необхідні для безперебійного функціонування складських приміщень.

Загальні проблеми логістики складських ресурсів обертаються навколо організації. Простіше кажучи, як можна досягти детального контролю над чимось великим, як склад? Потрібно бути в змозі точно визначити місце конкретного предмета інвентарю, піддону, на якому перевозився товар з товарним терміном дії, або вантажівку, що перевозила товар, пошкоджений під час відвантаження. Цей контроль є першорядним для безперебійного функціонування та стабільного доходу, але без професійних інструментів його практично неможливо досягти [4].

Недостатній простір для зберігання та неефективне використання наявних сховищ є поширеними проблемами на складах із поганим розташуванням приміщень. Погано налаштовані склади є головною причиною для занепокоєння керівників та власників через притаманний потенціал негативного впливу на прибуток. Необхідне оптимальне компонування як площі підлоги, так і вертикального простору, доступного для використання. Окрім максимального використання місця, хороший макет максимально використовує обладнання та робочу силу, доступність до всіх товарів та безпеку всіх предметів. Використання навантажувачів, які можуть досягти даху складу, дозволяє створити конфігурацію, яка максимально збільшує як горизонтальний, так і вертикальний простір. Ефективне використання місця є критичним фактором успіху при процесі складування .

Розроблювана система повинна частково допомогти у вирішенні цих проблем. Завдяки новітнім технологіям та значному розвитку в області штучного інтелекту можна створити систему, яка дозволить перекласти необхідність пошуку місця для нового товару з людини на нейронну мережу. Окрім того, однією з особливостей є можливість взаємодії з сучасними апаратними комплексами для автоматичного вимірювання розмірів нових одиниць, які приходять на склад. Таким чином мінімізується людський фактор та вплив на вибір правильної локації.

Від користувачів системи для правильної її роботи потрібно тільки вірно скласти повну мапу складу, з усіма можливими точками, полицями, де може зберігатися товар. Крім того, необхідно правильно вказати усі розміри згаданих вище місць. Будь яка помилка може призвести до некоректної роботи усієї системи і, відповідно, до матеріальних втрат.

Користувач системи зможе перевіряти результат роботи нейромережі, швидко шукати та отримувати інформацію про вже наявний товар, класифікувати його і загалом проводити всі ті операції, які доступні у багатьох вже існуючих системах керування складом.

1.2 Огляд існуючих рішень

Система управління складом (WMS) – це програмне забезпечення, яке допомагає контролювати та керувати щоденними операціями на складі. Програмне забезпечення WMS керує отриманням та вивезенням запасів, оптимізує вибір та доставку замовлень та надає рекомендації щодо поповнення запасів [5].

Раніше системи управління запасами складських приміщень могли забезпечувати лише прості функції, здебільшого лише інформацію про місце зберігання. В даний час широта функціональності WMS може сильно відрізнятися, від базового збору інформації про товар, упаковки та доставки, до складних програм, що координують передові взаємодії з пристроями матеріалів та управління територією.

Система управління складом може зменшити ймовірність помилок, які можуть виникнути при відвантаженні товару. Ця система також може допомогти компанії швидше виконувати замовлення та миттєво відстежувати замовлені товари на складі.

На ринку систем управлінь складом існує досить багато готових рішень з різним функціоналом та можливостями. Крім того, програмні та апаратні рішення в даній області є досить затребуваними, оскільки кожен склад є унікальним і потребує гнучкості в налаштуваннях від WMS системи.

Розглянемо декілька існуючих програмних рішень для формування власного уявлення про майбутній сервіс.

1.2.1 Система управління складом SphereWMS

Програмне забезпечення управління складами SphereWMS оптимізує робочий процес, напрямок та процеси на операційному рівні, щоб підвищити ефективність та знизити витрати на 3PL, розповсюдження, виконання, електронну комерцію та роздрібні операції.

Функції управління запасами включають поточну, транзитну, видимість на замовлення та контроль за рахунками інвентаря, місцями бункерів, номерами партії, ідентифікаторами піддону / справи та серійними номерами. Рішення надає інформацію в реальному часі, необхідну для відстеження кількості, місцезнаходження, стану та історії запасів товарів на складі.

SphereWMS дозволяє користувачам упорядкувати весь процес замовлення, інтегруючи бізнес-вимоги клієнтів безпосередньо в робочий процес та впроваджуючи рішення щодо виконання замовлень, пристосовані відповідно до їх потреб. SphereWMS може бути повністю інтегрований з існуючим ERP компанії та пропонує функціонування між платформами.

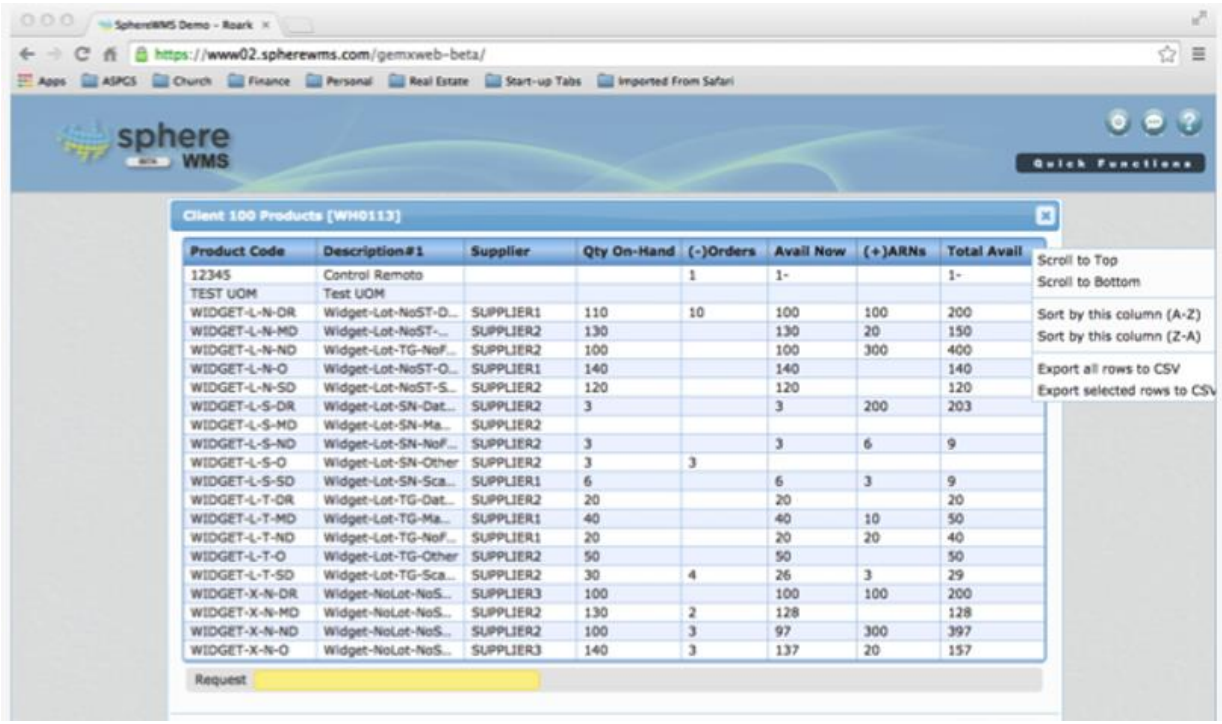


Рисунок 1.1. Інтерфейс SphereWMS

Можна виділити такі основні переваги системи:

– гнучкість у використанні;
– модульність.

Недоліки:

– складність інтерфейсу;
– деякі функції є досить громіздкими ;
– мала частота оновлень програми.

– досить велика ціна і для розширення функціоналу необхідно купляти додаткові модулі.

1.2.2 Система управління складом Logiwa

Logiwa WMS - це хмарне рішення для управління складами та виконання замовлень, побудоване на основі .NET. Рішення підтримує різні процеси складування, окремі групи продуктів та потреби різних секторів в рамках єдиної платформи. Він призначений для полегшення процесу від отримання на склад до відвантаження. Він підтримує багатосайтові компанії з даними в режимі реального часу і може використовувати такі технології, як RFID, штрих-кодування, світловий, голосовий та автоматизований MHS.

Logiwa WMS дозволяє користувачам здійснювати збирання, упаковку та доставку через мобільні додатки. Рішення допомагає користувачам збільшити продажі, синхронізуючи всі канали продажу в межах одного ланцюга поставок. Він забезпечує процес прийому та повернення, включаючи обробку замовлення та перехресне стикування.

Переваги:

– функціональність;
– гнучкість;
– кросплатформенність.

Недоліки:

– немає можливості зручно розміщувати новий товар;
– недоцільність для використання малими складами.

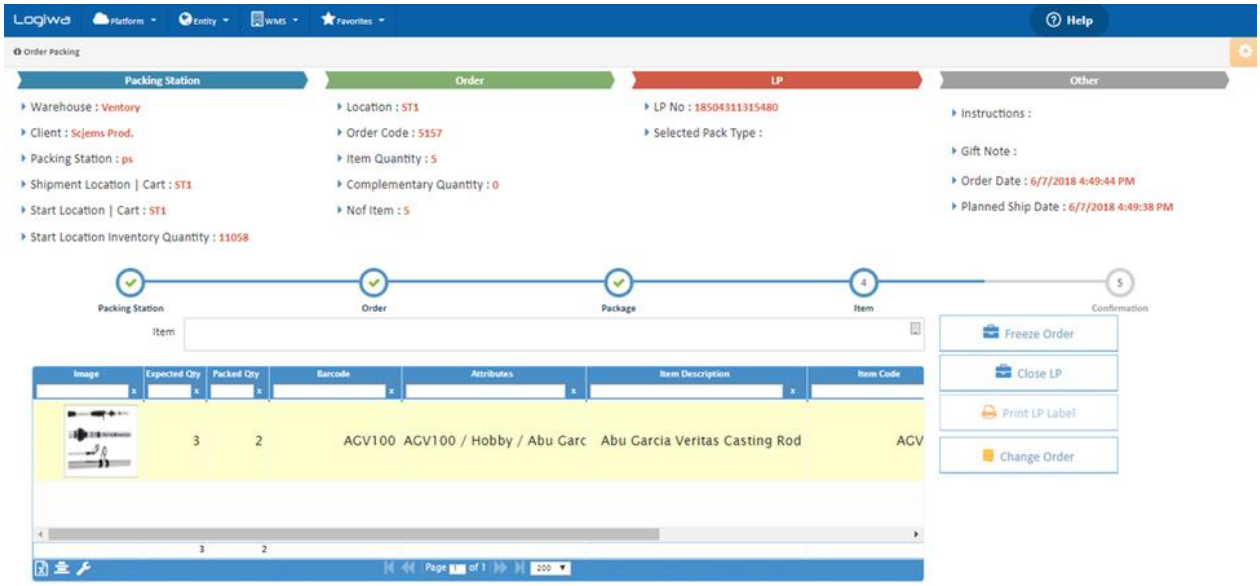


Рисунок 1.2. Інтерфейс користувача WMS системи Logiwa

1.2.3 Система управління складом Cadence WMS

Cadence WMS – це система управління складом, яка інтегрує складські операції з логістикою та реалізацією. Він орієнтований на середні та великі підприємства, на дистрибуційні та виробничі ринки.

Cadence керує складовими операціями для підприємств з кількома замовниками, постачальниками та складами. Він поєднує в собі єдине комплексне рішення видимості ланцюгів поставок, управління замовленнями, управління транспортом, виставлення рахунків за видами діяльності та управління складами.

Функціонал WMS включає прийом, спрямований спосіб, кілька способів вибору, доставку, набір, підрахунок циклу, звітність про працю та виставлення рахунків. Додаткові послуги включають управління складським приміщенням, розширене проектування складу, доступ до портал клієнтів та оповіщення. Обмін даними Cadence надає можливість інтеграції ERP та електронної комерції.

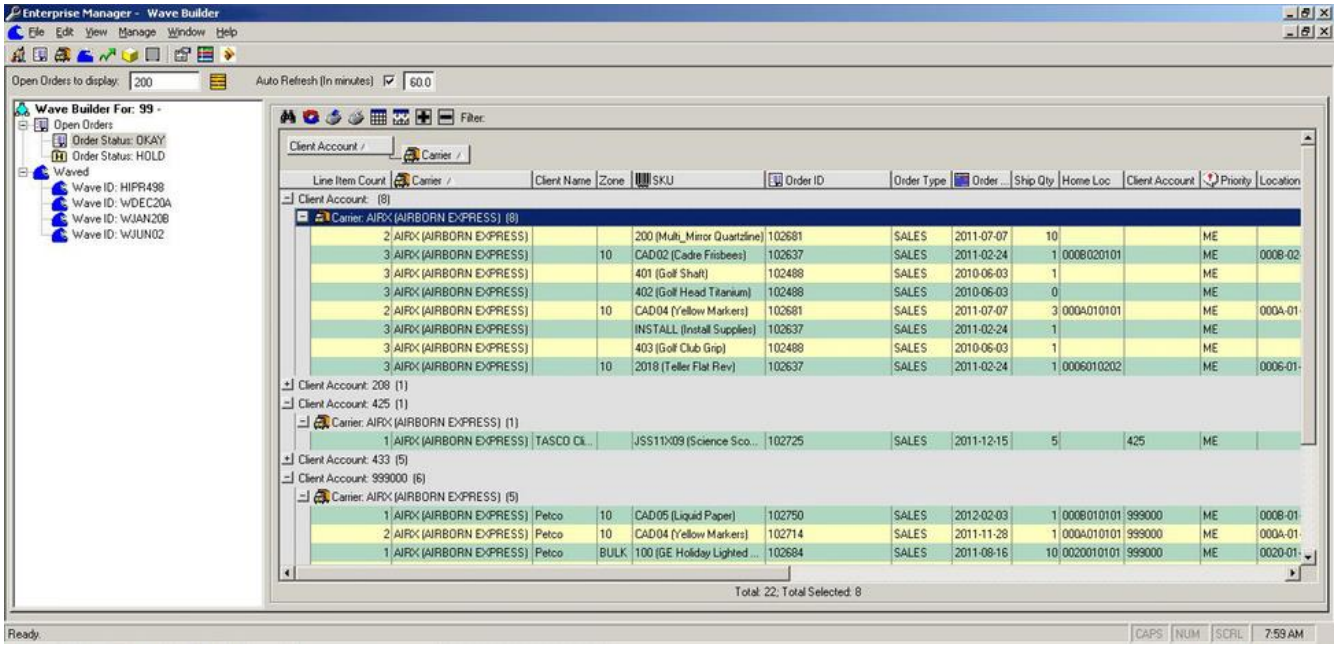


Рисунок 1.3. Інтерфейс користувача WMS системи Cadence

До переваг системи можна віднести такі як:

– простоту використання;

– функціональність.

Серед недоліків:

– застарілий інтерфейс користувача;

– велика ціна за користування.

1.2.4 Порівняння існуючих програмних рішень

В результаті проведеного аналізу сформуємо порівняльну таблицю 1.1, на якій представлені переваги та недоліки розглянутих продуктів.

Таблиця 1.1. Порівняння розглянутих комерційних рішень

Критерій	Sphere WMS	Logiwa	Cadence WMS
Функціональність	+	–	+
Зручність використання	–	–	–
Можливість розширення функціоналу	+	–	+
Можливість автоматично розподіляти товар	+	+	+

Закінчення таблиці 1.1

Відображення завантаженості та інформації про склад у реальному часі	+	+	–
Відображення статусу зарядних станцій в реальному часі	+	–	+
Підтримка мобільних платформ	+	+	–

Отже можна підсумувати, що основними критеріями для майбутньої сервісу є зручність інтерфейсу, функціональність та орієнтованість на процеси, що пов’язані безпосередньо із взаємодією товару та його розміщенням.

Спочатку сервіс буде мати тільки англомовний варіант, однак з часом будуть додаватися й інші локалізації. Ще однією перевагою є те, що для роботи веб-додатку немає різниці яка система у користувача, та не потрібно оновлювати клієнтські частини користувачів.

1.3 Постановка задачі

Метою роботи є створення інтелектуальної системи визначення оптимального місця для одиниці товару на складі з урахуванням його затребуваності та попиту. Для досягнення мети необхідно вирішити наступні задачі:

1. аналіз видів нейромереж, варіанти їх навчання;
2. розробка власного штучного інтелекту для реалізації поставленої мети;
3. розробка системи-сателіта, яка буде використовувати створену нейромережу;
4. розробка інтерфейсу користувача;
5. навчання та тестування нейромережі, перевірка роботи створеної системи.

Після аналізу існуючих програмних рішень було вирішено спроектувати систему, котра буде задовольняти наступні вимоги:

- розподіл ролей користувачів у системі для відображення тільки певного функціоналу, визначеного для певної ролі;
- обліковий запис, в якому будуть зберігатися персональні дані та налаштування користувача;
- доступ до фільтрів та зміна налаштувань повинні здійснюватися швидко, за рахунок кешування інформації;
- зручний конструктор для створення карти складського приміщення та оцифровки усіх параметрів складу;
- можливість зручно вводити та зберігати параметри нового товару, який поступає на склад;
- можливість автоматично знаходити відповідне місце на складі для нових товарів;
- можливість обрахунку оптимального часу на подолання вибраного маршруту із урахуванням часу підзарядки акумуляторів;
- зручний та інтуїтивно зрозумілий інтерфейс.

Призначенням розробки є допомога працівникам та менеджерам складу в процесі керування складським приміщенням, розміщенням нового товару, пошуком та відвантаженням уже наявного. Ключовою особливістю системи є повна автоматизація пошуку місця, де можна розмістити нових одиниць.

Також було вирішено розподілити ролі в системі таким чином:

- звичайний працівник;
- керівник складу;
- адміністратор.

Опис функціоналу, що доступний для звичайного працівника:

- перегляд мапи завантаженості складу;
- внесення в базу нових товарів, що прибули на склад;
- автоматичний пошук місця та, власне, розміщення товару
- відвантаження товару;
- виписка про наявні товари на складі.

Опис функціоналу, що доступний для керівника складу:

– перегляд мапи завантаженості складу;
– внесення в базу нових товарів, що прибули на склад;
– автоматичний пошук місця та, власне, розміщення товару;
– можливість редагування автоматично вибраного місця;
– можливість редагування та видалення товару з бази даних;
– можливість редагування мапи складу, зміна деяких параметрів складу (після відправки на погодження з адміністратором);
– виписка про наявні товари на складі;
– створення звітів по новим/відвантаженим товарам;
– редагування особистих даних.

Опис функціоналу, що доступний для адміністратора:

– перегляд мапи завантаженості складу;
– редагування мапи складу;
– повне створення, редагування, видалення можливих місць, де має знаходитись товар;
– внесення в базу нових товарів, що прибули на склад;
– автоматичний пошук місця та, власне, розміщення товару;
– можливість редагування автоматично вибраного місця;
– можливість редагування та видалення товару з бази даних;
– можливість редагування мапи складу, зміна деяких параметрів складу (після відправки на погодження з адміністратором);
– виписка про наявні товари на складі;
– створення звітів по новим/відвантаженим товарам;
– редагування особистих даних;
– редагування особистих даних інших користувачів системи.

Отже всі основні вимоги до системи визначено, можна перейти до вибору засобів реалізації.

Висновки по розділу

У першому розділі описано предметну область та чітко визначено мету розробки системи. Крім того, проведено аналіз існуючих трьох комерційних програмних рішень для предметної області для того, щоб визначити вимоги до системи та розуміти, чого саме бажають користувачі. Проведено порівняння цих систем, визначено переваги та недоліки цих рішень. В результаті проведеного аналізу сформовано вимоги до системи, за якими і буде вестися розробка.

РОЗДІЛ 2. АНАЛІЗ ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Аналіз нейронних мереж

2.1.1 Загальний опис нейронних мереж

Основна ідея нейронної мережі полягає в моделюванні (копіювання спрощеним, але досить вірним способом) безлічі щільно взаємопов'язаних клітин мозку всередині комп'ютера, щоб ви могли змусити його вивчати речі, розпізнавати закономірності та приймати рішення так, як це робить мозок. З цього випливає наступне: вам не потрібно програмувати її, щоб явно вчитися: вона вчиться сама, як і мозок.

Важливо зазначити, що нейронні мережі – це (в основному) програмне моделювання: їх створюють програмуючи звичайні комп’ютери. Нейронна мережа відрізняється від людського мозку точно так само, як комп’ютерна модель погоди відрізняється від реальних хмар, сніжинок чи сонячного світла. Комп'ютерне моделювання - це лише сукупність алгебраїчних змінних та математичних рівнянь, що пов'язують їх між собою. Іншими словами, числа, що зберігаються у полях, значення яких постійно змінюються. Вони нічого не означають для комп’ютерів, всередині яких працюють, - лише для людей, які їх програмують [6].

Справжні та штучні нейронні мережі. Строго кажучи, нейронні мережі, вироблені таким чином, називаються штучними нейронними мережами, щоб відрізняти їх від реальних нейронних мереж (колекцій взаємопов'язаних клітин мозку), які ми знаходимо всередині нашого мозку.

Структура нейронної мережі. У типовій нейронній мережі є від кількох десятків до сотень, тисяч, а то й мільйонів штучних нейронів, які називаються одиницями, розташованими в ряд шарів, кожен з яких з'єднується з шарами по обидва боки. Деякі з них, відомі як блоки введення, призначені для отримання різних форм інформації із зовнішнього світу, про які мережа намагатиметься дізнатися, розпізнати чи іншим чином обробити [7]. Інші підрозділи сидять на

протилежній стороні мережі та сигналізують, як вона реагує на інформацію, яку вона засвоїла; вони відомі як вихідні одиниці. Між вхідними і вихідними одиницями розташовані один або кілька шарів прихованих одиниць, які разом утворюють більшість штучного мозку. Більшість нейронних мереж повністю пов'язані, це означає, що кожен прихований блок і кожен вихідний блок підключені до кожного блоку в шарах з будь-якої сторони. Зв'язки між однією одиницею та іншою представлені числом, званим вагою, яке може бути як позитивним (якщо одна одиниця збуджує інше), так і негативним (якщо одна одиниця пригнічує або гальмує іншу). Чим більша вага, тим більший вплив однієї одиниці на іншу. (Це відповідає тому, як реальні клітини мозку спрацьовують одна одну через крихітні прогалини, які називаються синапсами)[8].

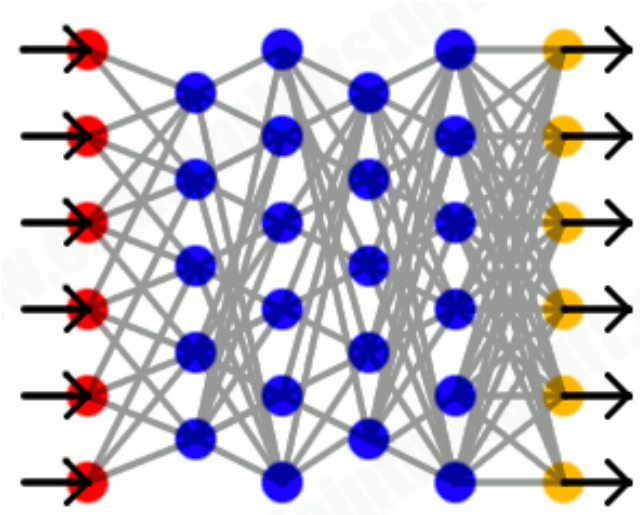


Рисунок 2.1. Принцип роботи нейромережі

На рис. 2.1 схематично показаний принцип роботи нейромережі. Повністю з'єднану нейронну мережу складають вхідні одиниці (червоний), приховані одиниці (синій) та вихідні одиниці (жовтий), причому всі одиниці з'єднані з усіма одиницями в шарах з обох боків. Вводи подаються зліва, активують приховані блоки посередині, а виходи подають праворуч. Міцність зв'язку між будь-якими двома одиницями поступово регулюється по мірі навчання мережі.

2.1.2 Типи нейронних мереж

Існує багато типів нейронних мереж. Їх можна класифікувати залежно від їх структури: потоку даних, використовуваних нейронів та їх щільності, шарів та фільтрів активації глибини тощо . Серед основних можна виділити наступні:

– нейронна мережа прямого поширення;
– багатошарові перцептронні нейронні мережі;
– згорткові нейронні мережі;
– рекурентні нейронні мережі.

1. Нейронна мережа прямого поширення

Найпростіша форма нейронних мереж, де вхідні дані рухаються лише в одному напрямку, проходячи через штучні нейронні вузли та виходячи через вихідні вузли [9]. Там, де приховані шари можуть бути відсутні вводяться та виводяться шари. Виходячи з цього, їх можна додатково класифікувати як одношарові або багатошарові подачі нейронних сіток вперед. Кількість шарів залежить від складності функції. Вони мають однонаправлене поширення вперед, але не мають зворотного поширення. Міцність зв’язків тут статична. Функція активації подається на введення, які множать на міцність. Для цього використовується класифікація функції активації або функції активації кроків. Наприклад: нейрон активується, якщо він перевищує поріг (зазвичай 0), і нейрон виробляє 1 як вихід. Нейрон не активується, якщо він нижчий за поріг (зазвичай 0), який вважається -1.

Переваги: менш складний, простий у проектуванні та обслуговуванні; швидкий; адекватно реагує на нечіткі дані.

Недоліки: не може використовуватись для глибокого навчання (через відсутність щільних шарів).

2. Багатошарові перцептронні нейронні мережі (MLP)

Найчастіше всього використовується у розпізнаванні мови, при машинному перекладі.

Точка входу до складних нейронних мереж, де вхідні дані переміщуються через різні шари штучних нейронів. Кожен окремий вузол з'єднаний з усіма нейронами в наступному шарі, що робить його повністю пов'язаною нейронною мережею. Присутні вхідні та вихідні шари, що мають кілька прихованих шарів, тобто щонайменше трьох або більше шарів. Він має двонаправлене поширення, тобто поширення вперед та назад. Вхідні дані множать на ваги і подають на функцію активації, а при зворотному розповсюдженні вони змінюються для зменшення втрат. Простими словами, ваги - це машинно засвоєні значення з нейронних мереж. Вони самостійно коригуються залежно від різниці між прогнозованими результатами та тренувальними входами. Функції нелінійної активації використовуються, а потім застосовується softmax-функція активації вихідного рівня [10].

Переваги: використовується для глибокого навчання (за рахунок наявності щільних повністю з’єднаних шарів та зворотнього відгуку)

Недоліки: порівняно складна для проектування та обслуговування; повільна (залежить від кількості прихованих шарів)

3. Згорткова нейронна мережа

Найчастіше всього використовується у обробці зображень, розпізнаванні мови, машинному перекладі.

Згорткові нейронні мережі містять тривимірне розташування нейронів замість стандартного двовимірного масиву. Перший шар називається згортковим. Кожен нейрон у згортковому шарі обробляє інформацію лише з невеликої частини зорового поля. Мережа розуміє зображення по частинах і може обчислити ці операції кілька разів, щоб завершити повну обробку зображення. Обробка передбачає перетворення зображення з RGB або HSI-масштабу в сірий. Подальша зміна значення пікселя допоможе виявити краї та зображення можна класифікувати на різні категорії [11].

Поширення є однонаправленим, коли нейромережа містить один або більше згорткових шарів з подальшим об'єднанням і двонаправленим, коли вихід шару згортки переходить до повністю пов'язаної нейронної мережі для

класифікації зображень. Фільтри використовуються для отримання певних частин зображення. У MLP вхід множать на ваги та подають на функцію активації. Згорткова нейромережа використовує RELU, а MLP використовує функцію нелінійної активації, за якою йде softmax [12].

Переваги: використовується для глибокого навчання з невеликими параметрами; менше параметрів, які можна вивчити порівняно з MLP.

Недоліки: порівняно складна для проектування та обслуговування; порівняно повільна (залежить від кількості прихованих шарів)

4. Рекурентні нейронні мережі

Найчастіше всього використовується у обробці тексту на зразок автоматичних пропозицій, перевірки граматики тощо; обробка тексту до мовлення; аналіз почуттів; машинний переклад.

Призначений для збереження виходу шару, повторювана нейронна мережа повертається на вхід, щоб допомогти передбачити результат шару. Перший шар, як правило, є нейронною мережею прямої подачі з подальшим періодичним шаром нейронної мережі, де деяка інформація, яку вона мала в попередньому кроці часу, запам'ятовується функцією пам'яті. В цьому випадку реалізоване розповсюдження. Він зберігає інформацію, необхідну для подальшого використання. Якщо прогноз невірний, для невеликих змін застосовується швидкість навчання. Отже, змушуючи його поступово збільшуватися до правильного прогнозування під час повернення.

2.1.3 Вибір нейронної мережі для системи автоматичного знаходження місця на складі

Для роботи системи, що буде сама приймати рішення щодо знаходження місця для нового товару, необхідно використовувати багатошарову нейронну мережу. Вона є більш гнучкою, володіє значно більшими можливостями за рахунок прихованих (проміжних) шарів [13].

2.1.4 Робота нейронної мережі

Як відомо, усі місця на складі мають певні розміри і поділені на певні частини. Тобто весь склад можна уявити як величезну кількість «комірок», у яких може зберігатися товар.

Крім того, у товару є свої параметри. Зазвичай це довжина, ширина, висота та вага. Тому саме ці параметри будуть вхідними для роботи мережі. На основі цих даних нейромережею проводитиметься підбір найбільш подібної «комірки», тобто місця, яке найточніше підходить під задані параметри. Після того, як штучний інтелект вибрав комірку, відбувається пошук такої вільної на території складу. У випадку успішного знаходження користувачу системи виводиться повідомлення з номером, або ж візуальне відображення, куди краще розмістити товар.

Формула нейронної мережі:

$$S = \sum_{i=1}^{n} X_i \cdot W_i.$$

В даній формулі X_i є вхідним сигналом, а W_i – значенням ваги [14].

2.2 Вибір засобу збереження даних

Для повноцінної роботи системи необхідно забезпечити належне зберігання усіх даних. Дані повинні зберігатися увесь час без залежності до живлення, яке подається на носій. Адже, як відомо оперативна пам’ять, де по замовчуванню усе зберігається, є енергозалежним носієм. Тобто навіть при короткочасній втраті напруги втрачається уся інформація, яка там зберігалася.

Серед ключових особливостей енергонезалежної пам’яті виділяється можливість зберігати файли, дані протягом тривалого часу. До енергонезалежних відносять такі носії як жорсткі диски, твердо тільні накопичувачі, карти пам’яті тощо.

Енергозалежним ж вважається такий тип пам’яті, який дозволяє зберігати інформацію тільки тоді, коли подається на нього напруга. Великою перевагою енергозалежної пам’яті є значно вища швидкість запису та

зчитування даних. Це дозволяє значно прискорити роботу з великими об’ємами інформації та зменшити час на їх опрацювання.

2.2.1 Енергозалежні засоби збереження даних

Для збереження тимчасової інформації в системі і швидкого доступу до неї було вирішено додати кешування. Кешування – це процес зберігання деяких даних у кеші. Кеш – це частина компонентів тимчасового зберігання, де зберігаються дані, щоб надалі ці дані можна було швидше отримати.

Для конкретного випадку з інтелектуальною системою найкращим варіантом став Redis.

Redis (розшифровується як Remote Dictionary Server) – це швидке сховище даних типу «ключ-значення» в пам'яті з відкритим вихідним кодом для використання в якості бази даних, кеша, брокера повідомлень або черги. Redis забезпечує час відгуку на рівні частки мілісекунди і дозволяє додаткам, що працюють в режимі реального часу, виконувати мільйони запитів в секунду. Redis широко застосовується для кешування, управління сеансами, аналітики в режимі реального часу, роботи з геопросторовими даними, підтримки служб таксі, чатів і сервісів обміну повідомленнями, потокової передачі мультимедіа.

Всі дані в Redis зберігаються в пам'яті, а не на дисках або твердотільних накопичувачах, як в інших базах даних. Оскільки Redis, як і інші сховища даних в пам'яті, не потребує доступі до диска, це виключає затримки, пов'язані з пошуком, і забезпечує доступ до даних за мікросекунди. У число можливостей Redis входить підтримка різноманітних структур даних, забезпечення високої доступності, створення скриптів Lua, проведення транзакцій, постійне зберігання даних на диску і підтримка кластерів. Все це спрощує створення додатків, що працюють в режимі реального часу в масштабі всього Інтернету.

2.2.2 Енергонезалежні засоби збереження даних

Для того щоб система була стійкою до відмов та мала можливість відновити усю інформацію, яка могла зберігатися на енергозалежній пам’яті (тобто в кеші) необхідно реалізувати механізм записування цих даних на енергонезалежний ресурс, тобто базу даних.

В якості бази даних було обрано рішення від Microsoft – MS SQL Server 2016. Це є реляційна SQL база даних. Microsoft SQL Server – це система управління реляційними базами даних, яка підтримує широкий спектр програм для обробки транзакцій, бізнес-аналітики та аналітики в корпоративних IT-середовищах. Microsoft SQL Server є однією з трьох лідируючих на ринку технологій баз даних, поряд з Oracle Database та IBM DB2.

Як і інше програмне забезпечення, Microsoft SQL Server побудований на базі SQL, стандартизованої мови програмування, яку адміністратори бази даних та інші IT-фахівці використовують для управління базами даних та запиту даних, які вони містять. SQL Server пов'язаний з Transact-SQL (T-SQL), реалізацією SQL від Microsoft, яка додає набір стандартних розширень програмного забезпечення до стандартної мови [15].

Як і інші технології СУБД, SQL Server в основному побудований навколо структури на основі рядків, яка з'єднує пов'язані елементи даних в різних таблицях один з одним, уникаючи необхідності надмірно зберігати дані в декількох місцях в базі даних. Реляційна модель також забезпечує референтну цілісність та інші обмеження цілісності для підтримки точності даних. Ці перевірки є частиною більш широкого дотримання принципів атомності, узгодженості, ізоляції та довговічності, спільно відомих як властивості ACID, і розроблені таким чином, щоб гарантувати надійну обробку операцій з базами даних.

Основним компонентом Microsoft SQL Server є SQL Server Database Engine, який контролює зберігання, обробку даних та безпеку. Він включає реляційне ядро, яке обробляє команди та запити, і механізм зберігання даних,

який управляє файлами баз даних, таблицями, сторінками, індексами, буферами даних та транзакціями. Збережені процедури, тригери, перегляди та інші об'єкти бази даних також створюються та виконуються Механіком баз даних [16].

В ядрі бази даних знаходиться операційна система SQL Server або SQLOS. SQLOS обробляє функції нижчого рівня, такі як управління пам'яттю та введення-виведення, планування роботи та блокування даних, щоб уникнути конфліктних оновлень. Мережевий рівень інтерфейсу розташовується над Database Engine і використовує табличний протокол потоку даних Microsoft для полегшення взаємодії запитів і відповідей із серверами баз даних. І на рівні користувача, DBA та розробники SQL Server записують T-SQL заяви для створення та зміни структур баз даних, маніпулювання даними, здійснення захисту безпеки та резервного копіювання баз даних, серед інших завдань. Схему з архітектурою SQL сервера показано на рис. 2.2.

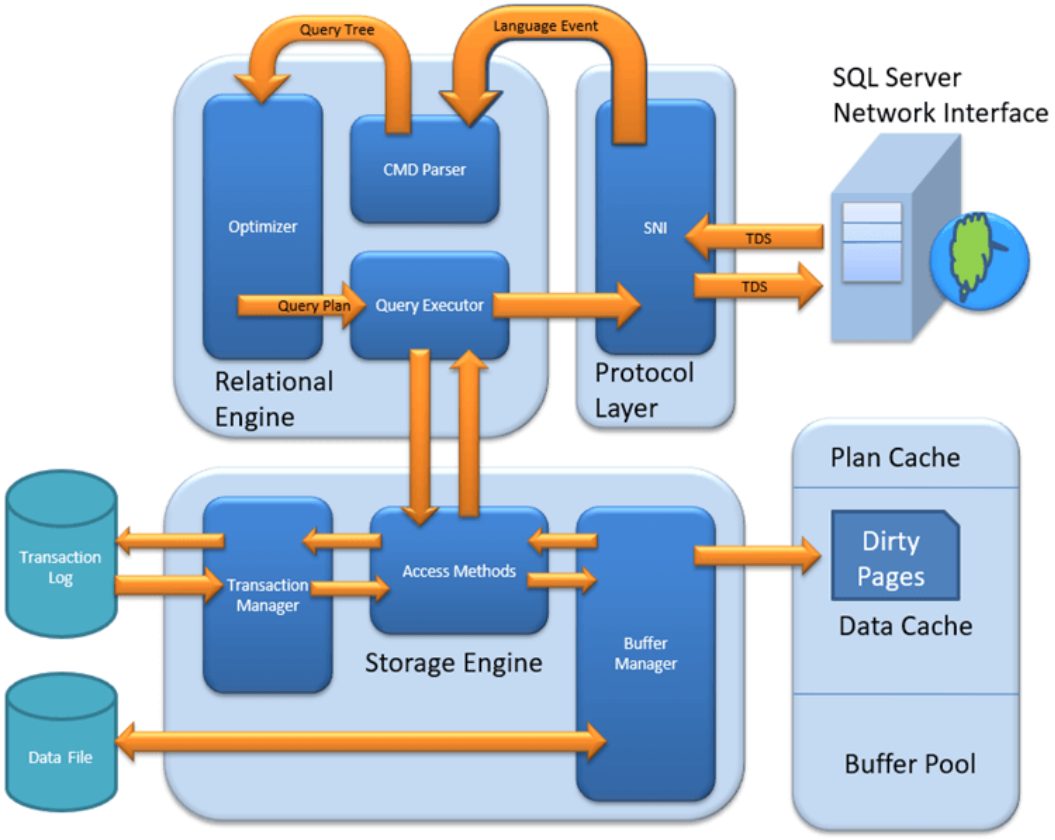


Рисунок 2.2. Архітектура MS SQL Server

2.3 Вибір платформи розробки системи

Вибір платформи для розробки майбутньої системи є надзвичайно важливим етапом, оскільки від нього залежить подальша робота, швидкодія, гнучкість та функціональність створюваного продукту. Тому до цього потрібно відноситись з усією відповідальністю.

2.3.1 Фреймворк ASP.NET

В якості основної платформи для побудови веб-застосунку було обрано ASP.NET MVC.

ASP.NET – платформа для веб-розробок, що надається Microsoft. Вона використовується для створення веб-додатків [17]. ASP.NET вперше вийшов у 2002 році.

Існує маса вагомих причин використовувати ASP.NET при розробці веб-сайту чи програми. Висока швидкість, низька вартість та широка підтримка мови є одними з найбільш значущих переваг. ASP.NET вбудований у звичне середовище сервера Windows, вимагає менших налаштувань та конфігурації, ніж інші платформи веб-розробки, які повинні бути встановлені та налаштовані окремо. Популярність ASP.NET дозволяє легко знайти Інтернет-ресурси та кваліфікованих розробників.

Веб-сайти та програми, створені за допомогою ASP.NET, можуть бути швидшими та ефективнішими, ніж, наприклад, створення веб-сайтів за допомогою PHP. Програми ASP.NET компілюються, що означає, що код переводиться в об'єктний код, який потім виконується [18]. Цей процес компіляції займає невелику кількість часу, але відбувається лише один раз. Після компіляції код може бути виконаний знову і знову платформою .NET дуже швидко.

MVC – це шаблон проектування. Він розділяє додаток на три основні компоненти: "Модель", "Представлення" та "Контролер". MVC є найбільш часто використовуваною галузевою основою веб-розробки.

Модель – простий клас. Це форма даних. Модель містить бізнес-логіку. Контролер та View можуть отримати доступ до моделі. Модель допомагає передавати дані з контролера для перегляду та перегляду до контролера. За допомогою моделі ми відображаємо дані на сторінці перегляду.

Представлення – це інтерфейс користувача. Воно використовується для відображення всіх даних за допомогою моделі.

Контролер є серцем MVC і він обробляє запит користувача. Це простий клас. Контролер може отримати доступ до моделі та передавати дані для перегляду за допомогою моделі. Ми можемо передавати дані між контролером і переглядати за допомогою *ViewData*, *TempData* та *ViewBag*. Контролер є проміжним між моделлю та видом.

2.3.2 IIS веб-сервер

Для того, щоб створений за допомогою ASP.NET додаток був доступний його необхідно розмістити на сервері. В якості сервера у розроблюваній системі буде використовуватися IIS веб-сервер.

IIS - це програмний пакет для веб-сервера, призначений для Windows Server [20]. Він використовується для розміщення веб-сайтів та іншого вмісту в Інтернеті.

Інформаційні послуги Інтернету Microsoft надають графічний інтерфейс користувача (GUI) для управління веб-сайтами та пов'язаними з ними користувачами. Він надає наочні засоби створення, налаштування та публікації сайтів в Інтернеті. Інструмент IIS Manager дозволяє веб-адміністраторам змінювати параметри веб-сайтів, такі як сторінки за замовчуванням, сторінки помилок, налаштування журналу, налаштування безпеки та оптимізація продуктивності.

IIS може обслуговувати як стандартні веб-сторінки HTML, так і динамічні веб-сторінки, такі як програми ASP.NET та сторінки PHP. Коли відвідувач отримує доступ до сторінки на статичному веб-сайті, IIS просто надсилає HTML і пов’язані з ними зображення в браузер користувача. Коли

доступ до сторінки на динамічному веб-сайті, IIS запускає будь-які програми та обробляє будь-які сценарії, що містяться на цій сторінці, а потім надсилає отримані дані в браузер користувача.

IIS - популярний варіант для комерційних веб-сайтів, оскільки пропонує безліч вдосконалених функцій і підтримується Microsoft. Однак для цього також потрібна комерційна ліцензія, і ціна збільшується залежно від кількості користувачів. Тому Apache HTTP Server, який з відкритим кодом та безкоштовний для необмежених користувачів, залишається найпопулярнішим програмним забезпеченням для веб-серверів.

2.3.3 ML.NET

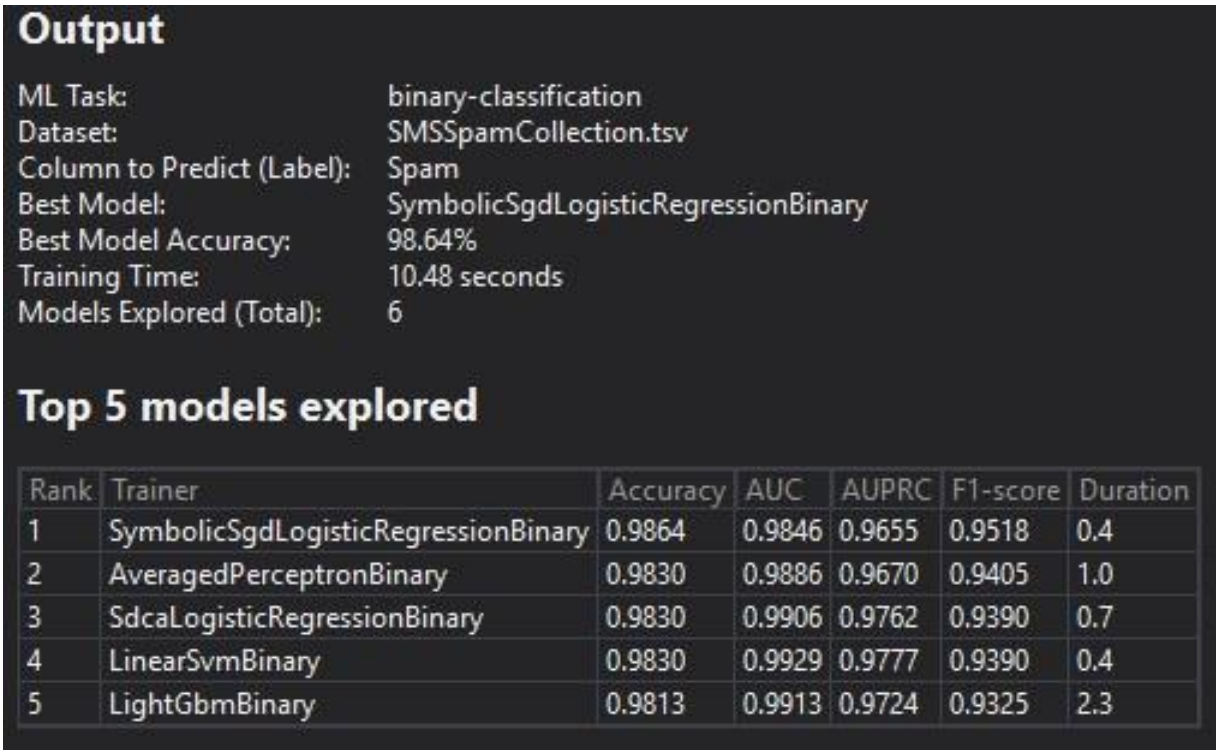
ML.NET – це вільна бібліотека машинного навчання для мов програмування C# і F#. ML.NET включає перетворення для таких інженерних функцій, як створення n-грамів, і може навчатися для вибору за двійковою класифікацією, багатокласовою класифікацією та регресійними завданнями. Також в майбутні версії можуть бути включені додатковий функціонал ML, такі як системи виявлення аномалій та рекомендацій, а інші підходи, такі як глибоке навчання.

ML.NET дозволяє розробникам додавати модельно-орієнтовану аналітику, яка базується на Machine Learning до існуючих .NET проектів. Фреймворк побудований під .NET Core та .NET Standard, що зберігає можливість робити кросплатформенні додатки.

Розробники можуть навчити модель машинного навчання на тестових датасетах, що дозволить користуватися нейромережею навіть за відсутності з’єднання з мережею Інтернет. Це означає, що розробникам не потрібно мати багато досвіду роботи з Data Science, щоб використовувати фреймворк.

Наприклад, для того щоб створити просту нейронну мережу з машинним навчанням, розробнику потрібно просто розробити модель, яка буде використовуватись як цільова для прогнозування. Тобто, створивши таку модель, та навчивши на тестових даних, можна дуже швидко отримати готову

робочу нейронну мережу, яка може прогнозувати результат з досить високою точністю. На рис. 2.3 відображено приклад роботи ML.NET.



Output

ML Task:	binary-classification
Dataset:	SMSSpamCollection.tsv
Column to Predict (Label):	Spam
Best Model:	SymbolicSgdLogisticRegressionBinary
Best Model Accuracy:	98.64%
Training Time:	10.48 seconds
Models Explored (Total):	6

Top 5 models explored

Rank	Trainer	Accuracy	AUC	AUPRC	F1-score	Duration
1	SymbolicSgdLogisticRegressionBinary	0.9864	0.9846	0.9655	0.9518	0.4
2	AveragedPerceptronBinary	0.9830	0.9886	0.9670	0.9405	1.0
3	SdcaLogisticRegressionBinary	0.9830	0.9906	0.9762	0.9390	0.7
4	LinearSvmBinary	0.9830	0.9929	0.9777	0.9390	0.4
5	LightGbmBinary	0.9813	0.9913	0.9724	0.9325	2.3

Рисунок 2.3. Приклад роботи ML.NET

Крім того, фреймворк ML.NET відрізняється швидкістю роботи та швидкістю навчання. Звичайно, на останню впливає такий фактор, як розмір навчального датасету, однак все ж результуючі цифри є меншими за інші варіанти написання нейронних мереж.

2.4 Вибір засобу представлення системи

В попередніх розділах мова йшла про створення серверної частини додатку. Однак, інформація потребує виведення на екран у зручній для користувача формі. Тому для цього створюються представлення, які взаємодіють з серверною частиною додатку і виводять дані у потрібному вигляді.

Представлення будуть написані на HTML з широким використанням стилів CSS, мови програмування JS а також різних фреймворків.

2.4.1 Фреймворк Bootstrap

Bootstrap – це вільна та відкрита платформа для розробки представлень для створення веб-сайтів та веб-додатків. Фреймворк Bootstrap побудована на HTML, CSS та JavaScript (JS) для полегшення розробки гнучких мобільних сайтів та додатків [21].

Гнучкий дизайн дає можливість веб-сторінці чи додатку визначати розмір та орієнтацію екрана відвідувача та автоматично адаптувати дисплей відповідно. Перший підхід для мобільних пристроїв передбачає, що смартфони, планшети та мобільні додатки, що стосуються конкретних завдань, є основним інструментом для роботи і відповідає вимогам цих технологій у дизайні.

Bootstrap включає компоненти користувальницького інтерфейсу, макети та інструменти JS разом із основою для реалізації. Програмне забезпечення доступне заздалегідь складеним або як вихідний код.

2.4.2 Фреймворк Angular

AngularJS – це фреймворк з відкритим кодом Model-View-Controller, який схожа на JavaScript.

AngularJS – це, мабуть, один з найпопулярніших сучасних веб-фреймворків, доступних сьогодні. Цей фреймворк використовується для розробки в основному односторінкових додатків. Цю основу розробила група розробників з Google.

Через повну підтримку Google та ідеї широкого форуму спільноти фреймворк завжди оновлюється. Крім того, він завжди включає новітні тенденції розвитку на ринку.

Angular має наступні ключові особливості:

– MVC – фреймворк побудований на відомій концепції MVC (Model-View-Controller). Це шаблон проектування, яка використовується у всіх сучасних веб-додатках. Цей шаблон заснований на розбитті шару бізнес-логіки, рівня даних та шару презентації на окремі розділи.

Поділ на різні секції робиться так, щоб кожним можна було легше керувати.

- Прив’язка моделі даних - вам не потрібно писати спеціальний код для прив’язки даних до елементів керування HTML. Це може зробити Angular, просто додавши кілька фрагментів коду.
- Можливість писати менше коду – з Angular можна обходитись меншою кількістю коду, який потрібно написати для маніпуляції з DOM.

Висновки по розділу

В даному розділі розглянуто і проаналізовано нейромережі, їх можливі типи. Описано алгоритм роботи нейромережі для інтелектуальної системи управління складом. Крім того, визначено технології, за допомогою яких буде написана система. Мова програмування – C#, серверна частина на технології ASP.NET MVC. Для зберігання даних буде використовуватись MS SQL Server.

РОЗДІЛ 3. РОЗРОБКА АЛГОРИТМІЧНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Архітектура програмного забезпечення

3.1.1 Трирівнева архітектура системи

Після аналізу вимог до системи керування складом було обрано трирівневу клієнт-серверну архітектуру. Така архітектура складається з наступних частин:

- сервер баз даних;
- сервер застосунку;
- клієнтська частина.

Також були вибрані наступні технології для реалізації спроектованої архітектури:

- мова програмування C# як основний інструмент розробки;
- платформа .NET Framework;
- Internet Information Services як сервер веб-додатків;
- Microsoft SQL Server 2016 в якості СУБД;
- мова програмування JavaScript для реалізації клієнтської частини;
- ASP.NET MVC 5 у якості фреймворку для створення веб-застосунків, який реалізує шаблон проектування Модель-Представлення-Контролер (Model-View-Controller)
- Entity Framework версії 6 у якості технології об’єктно–реляційного проектування даних, для роботи з інформацією, що зберігається в БД:
- формат даних JSON для обміну інформацією між клієнтською частиною та серверною;

Трирівнева архітектура дозволяє чітко розділити частини, які відповідають за збереження даних, їх обробку та вивід користувачу. Зазвичай такий тип архітектури використовують у випадку, коли система має бути розрахована на велику кількість клієнтів.

Трирівневу клієнт серверну архітектуру використовують для розділення частин, що відповідають за збереження, обробку та відображення даних [6]. Таку архітектуру доцільно використовувати, коли планується будувати систему, що розрахована на велику кількість клієнтів. Даний тип архітектури має такі основні переваги:

– відсутність дублювання коду програми-серверу і програм-клієнтів;
– оскільки всі обрахунки здійснюються на сервері, то знижуються вимоги до потужності комп’ютерів-клієнтів;
– дані зберігаються на сервері баз даних, котрий більш захищений ніж комп’ютери клієнти;
– на сервері додатків простіше організувати контроль доступу до даних, щоб дозволяти доступ тільки тим клієнтам, що мають відповідні права.

Архітектура системи представлена на рис. 3.1.

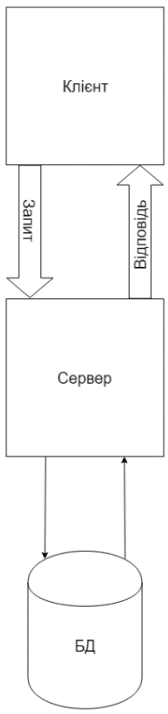


Рисунок 3.1. Архітектура системи

База даних дозволяє зберігати дані і виноситься на окремий рівень та реалізується засобами СУБД. Підключення до цього компоненту забезпечується тільки з рівня серверу додатків.

Сервер розміщений на іншому рівні і вся бізнес-логіка зосереджена саме на ньому. Тобто сервер відповідає за правильність обробки інформації яка надходить як з клієнтської сторони, так і зі сторони баз даних.

Клієнт – це те, що бачить користувач. Цей рівень не має прямих зв’язків із базою даних, не може містити в собі бізнес–логіку і зберігати стан. На цей рівень зазвичай виноситься тільки примітивна логіка, така як: авторизація, перевірка значень, що вводяться на допустимість та відповідність формату, нескладні операції з малою кількістю даних (сортування, угрупування, підрахунок значень), що вже були завантажені на клієнтський пристрій.

На серверах встановлена мережева операційна система та потужне апаратне забезпечення. Основними ролями сервера є:

– управління трафіком;
– забезпечення централізованого захисту інформації;
– надання інформації клієнтам;
– доступ до спільних пристроїв та файлів;
– виконання обчислень.

Такий тип розподілу ролей називають клієнтською (front-end) та серверною (back-end) обробкою. Мережі моделі клієнт–сервер можуть працювати з мікрокомп’ютерами та мейнфреймами, і ця гнучкість з врахуванням достатньо низької вартості є однією з основних переваг даного типу архітектури.

Також зазвичай на сервері встановлена спеціалізована операційна система. Серверна операційна система – це спеціальне програмне забезпечення, яке використовується в якості платформи для запуску багатокористувацьких комп'ютерних програм, програм, мережевих програм, а також для вирішення важливих обчислювальних задач для бізнесу.

Зокрема серверна операційна система в більшій мірі базується на управлінні ресурсами, клієнтська в свою чергу на виконання різних процесів з максимальною ефективністю та швидкістю. Також серверні операційні системи

можуть функціонувати як в якості програмного забезпечення сервера, так і клієнта. Для виконання різноманітних обов’язків серверна операційна система повинна працювати з максимально можливою швидкістю, і це реалізується за допомогою багатопроцесорності, багатопоточності та багатозадачності.

При багатопроцесорній роботі система може краще виконувати задачі за рахунок того, що навантаження перекладається на багато паралельних процесорів.

Багатопроцесорну обробку розділяють на симетричну та асиметричну. При симетричній обробці будь який обчислювальних процес може бути доручений будь якому в даних момент вільному процесору. А при асиметричній навантаження розподіляється так, щоб деяка підмножина процесорів обслуговувала основну операційну систему, а інша працювала з застосуваннями.

Багатопоточність – це одна з основних особливостей мережевої операційної системи. За рахунок повного використання процесорного часу даний метод є дуже ефективним. Принцип роботи полягає в тому, що процесор працює з великою швидкістю, яка вимірюється в тактах, причому ці такти виконують незалежно від занятості процесора. Також кожен процес поділяється на незалежні один від одного потоки, які керуються операційною системою.

При виконанні системою функції багатозадачності час процесора дається кожному процесу окремо, хоча користувачу здається, що ці процеси виконуються паралельно. Це відбувається завдяки високій швидкості виконання обчислювальних процесів процесором і також властивістю перемішувати виділені інтервали часу. Даний принцип ефективний як засіб підвищення продуктивності в роботі. Ще більша ефективність буде досягнута при реалізації даного принципу на пристрої клієнта, що дозволить ще краще керувати взаємодією сервера з клієнтом і досягнути цим ще більшої продуктивності на відміну від випадку, коли сервер та клієнт в деякий момент часу обробляють окремо або разом єдину задачу.

3.1.2 Реалізація моделі баз даних

Для того, щоб створити базу даних з таблицями використовувався підхід Code First ORM фреймворка Entity Framework. Мається на увазі, що перш за все в мові програмування C# були створені моделі зі зв’язками. Після цього, при першому зверненні з коду до об’єктів створилися таблиці в базі даних. Процес створення таблиць в базі даних схематично наведений на рис. 3.2.

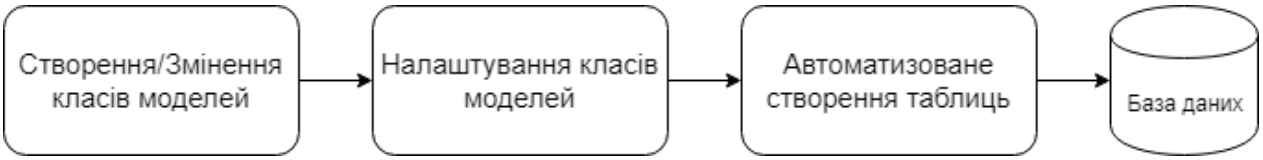


Рисунок 3.2. Створення таблиць через Entity Framework

Нижче наведена структура таблиць бази даних, схема бази даних наведена у додатку графічного матеріалу.

Таблиця 3.1. Опис таблиці «Користувач» (*User*)

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
UserId	Унікальний ідентифікатор користувача системи	*INT*	11	X	
UserName	Логін користувача	*VACRCHAR*	100		
Password	Пароль користувача	*VACRCHAR*	100		
FirstName	Ім’я користувача	*VACRCHAR*	100		
LastName	Прізвище користувача	*VACRCHAR*	100		
Email	Електронна пошта	*VACRCHAR*	100		

Закінчення таблиці 3.1

Phone	Контактний номер телефону	*VACRCHAR*	20		
FacilityId	Ідентифікатор складу	*INT*	11		X

У таблиці «Користувач» зберігаються дані користувачів системи, тобто працівників складу. Створення нового працівника проводиться адміністратором. Він заповнює усі дані, контактну інформацію та призначає працівнику ролі.

Таблиця 3.2. Опис таблиці «Роль» (*Role*)

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
RoleId	Унікальний ідентифікатор ролі	*INT*	11	X	
RoleName	Назва ролі	*VACRCHAR*	100		

У таблиці «Роль» зберігаються ролі, які використовуються у системі, оскільки доступ до різних модулів системи буде обмежуватися ролями, які призначені користувачеві. Перевірка відбуватиметься в контролерах і, за відсутності певної ролі, користувача буде повідомлено та перенаправлено на сторінку входу.

Таблиця 3.3. Опис таблиці «Товар» (*Item*)

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
ItemId	Унікальний ідентифікатор товару	*INT*	11	X	

Закінчення таблиці 3.1

ItemKey	Ключ товару	*VACRCHAR*	100		
Description	Опис товару	*VACRCHAR*	100		
UpcCode	Загальний код UPC	*VACRCHAR*	14		
InnerUpcCode	Код UPC на території складу	*VACRCHAR*	14		
ItemWidth	Ширина	*DECIMAL*	18,2		
ItemLength	Довжина	*DECIMAL*	18,2		
ItemHeight	Висота	*DECIMAL*	18,2		
ItemWeight	Вага	*DECIMAL*	18,2		
DemandFactor	Коефіцієнт попиту	*INT*	11		
LocationId	Ідентифікатор місця розташування	*INT*	11		X
ItemStatusId	Ідентифікатор статусу товару	*INT*	11		X

У таблиці «Товар» міститься повна інформація про одиницю товару, що надходить та перебуває на складі. Особливо важливими полями для заповнення в таблиці при додаванні є фактичні габарити товару, його вага та унікальний внутрішній UPC код. Останній, як і *PickLocationId* та *ItemStatusId* заповнюється після видачі та підбору локації для товару, а використовується задля швидкого та зручного пошуку, сканування та зміни статусу для подальшого відвантаження.

Таблиця 3.4. Опис таблиці «Склад» (*Facility*)

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
FacilityId	Унікальний ідентифікатор Складу	*INT*	11	X	
FacilityCode	Скорочена назва складу (код)	*VACRCHAR*	12		
FacilityName	Назва складу	*VACRCHAR*	100		
FacilityAddress	Адреса складу	*VACRCHAR*	100		

Таблиця «Склад» містить інформацію про різні склади, що можуть використовуватися однією компанією. Тобто створювана система управління орієнтована також на підтримку декількох складів. Крім того, в таблиці «Користувач» можна задавати склади, в яких він має право працювати.

Таблиця 3.5. Опис таблиці «Місце» (*Location*)

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
LocationId	Унікальний ідентифікатор товару	*INT*	11	X	
Location	Ключ товару	*VACRCHAR*	100		
LocationCode	Опис товару	*VACRCHAR*	100		
FacilityId	Ідентифікатор складу	*INT*	11		X
SizeId	Ідентифікатор розміру	*INT*	11		X

Таблиця «Місце» містить інформацію про усі можливі місця розміщення товарів та складських одиниць. Оскільки вона має код місця, то по ньому може зручно реалізовуватись навігація, на кшталт як це відбувається всередині великих приміщень. Саме адміністратор заповнює інформацію про локації, і всі дані з цієї таблиці складають так звану карту складу.

Таблиця 3.6. Опис таблиці «Розмір» (*Size*)

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
SizeId	Унікальний ідентифікатор розміру	*INT*	11	X	
SizeCode	Код розміру	*VACRCHAR*	100		
Height	Максимальна висота	*DECIMAL*	18,2		
Width	Максимальна ширина	*DECIMAL*	18,2		
Length	Максимальна довжина	*DECIMAL*	18,2		
MaxWeight	Максимальна вага	*DECIMAL*	18,2		

В таблиці розмірів зберігатимуться усі можливі розміри локацій. Саме з цієї таблиці нейромережа буде обирати найбільш ефективний розмір і далі система буде шукати вільну локацію для розміщення товару.

Таблиця 3.7. Опис таблиці «Статус» (*Status*)

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
StatusId	Унікальний ідентифікатор статусу	*INT*	11	X	

Закінчення таблиці 3.1

StatusName	Назва статусу	*VACRCHAR*	50		
StatusCode	Код статусу	*VACRCHAR*	50		

Таблиця «Статус» зберігатиме інформацію про усі можливі статуси товару під час перебування на складі.

Таблиця 3.8. Опис зв’язуючої таблиці користувачів та ролей (User_Role)

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
User_RoleId	Унікальний ідентифікатор	*INT*	11	X	
UserId	Ідентифікатор користувача	*INT*	11		X
RoleId	Ідентифікатор ролі	*INT*	11		X

3.1.3 Реалізація серверної частини

Оскільки як технологію додатку в другій частині роботи було вибрано ASP.NET MVC, то додаток буде розроблюватись на основі шаблону проектування MVC. Тобто серверний додаток буде складатися з трьох основних частин: контролерів, моделей та представлень.

Представлення відображаються браузером і саме з представленнями взаємодіє користувач при роботі з системою. Генеруються вони класами контролерів, тобто коли з веб-браузера відсилається запит на сервер, викликається дія (*Action*) в класі контролера.

Для прикладу візьмемо клас контролера, який керує користувачами, тобто містить у собі дії для виведення, додавання, зміни та видалення користувачів. Це контролер *UsersController*. Всі контролери крім Web API

контролерів в проекті наслідуються від *BaseController*, який в свою чергу наслідується від класу бібліотеки MVC Controller.

Клас *UserController* складається з 8 *Actions*, які відповідають за обробку вхідного запиту та надсилання правильної відповіді сервера. Для функціональної роботи з користувачами діями є:

– відображення головної сторінки (*Index*);
– відображення сторінки перегляду деталей користувача (*Details*);
– додавання нових користувачів (*Add*);
– редагування користувачів (*Edit*);
– видалення користувачів (*Delete*).

В контролерах широко застосовується перевантаження методів, тобто може міститись 2 методи з однаковою назвою, при цьому вони можуть реалізовувати різний функціонал. Зазвичай вони відрізняються вхідними параметрами та типом запиту, який на них має надходити. Наприклад, в контролері *UsersController* є дві дії для додавання користувача (*Create*). Перша не приймає жодних параметрів та оброблює Get запит. Друга ж приймає як параметр клас моделі «*User*», тобто екземпляр класу з заповненими полями, що надходить з представлення. Окрім того, це Post запит. Post запити відрізняються від Get запитів тим, що параметри в них передаються не в адресному рядку у відкритому вигляді, а в самому тілі запиту. Саме тому Post запити є більш безпечними і їх прийнято використовувати для передачі великої кількості параметрів. Також, максимальна довжина Get параметра становить 255 символів, тоді як Post необмежений.

Ще один метод, який є в контролері – це метод *Dispose*, що призначений для звільнення ресурсів, які використовуються додатком. В .NET Framework реалізований збірник «сміття» Garbage collector, однак логіка його роботи не передбачає звільнення деяких ресурсів, які можуть використовуватись в момент зачистки. Для того розробник повинен реалізовувати Dispose метод, в якому

жорстко прописана очистка необхідних об’єктів з оперативної пам’яті. Це дозволяє уникнути проблем з використанням пам’яті та витоку пам’яті.

Виклик методів виконується за допомогою підсистеми маршрутизації і модуля *UrlRoutingModule*. Маршрутизація ASP.NET дозволяє використовувати URL адреса, які не співвідносяться з файлами на сервері. Шляхи зберігаються в статичному класі *RouteTables* у властивості *Routes*. Наприклад, для вибраного вище контролера маршрут в адресному рядку веб переглядача буде виглядати наступним чином – «http://localhost:57229/Users/Index».

При роботі з даними використовується об’єкт контексту даних *WmsDbContext*. Це клас, згенерований Entity Framework при створенні моделей з бази даних. Через контекст витягуються колекції через фільтри, які попередньо вводить розробник. Так як робота з базою проводиться через вбудовані класи перечислень і списків, доступні дії додавання, видалення та редагування елементів.

Також, для роботи з даними реалізовані шаблони проектування UnitOfWork та Repository. UnitOfWork працює зі списком об’єктів, слідкує які були змінені, додані або видалені, координує збереження змін та вирішує проблеми паралельної роботи з даними. Шаблон проектування Reporitory представляє собою посередника між доменною моделлю та розподілом даних, використовуючи інтерфейс, який схожий зі стандартним інтерфейсом колекцій.

Окрім контролерів, які повертають вихідний код для його відображення безпосередньо на веб переглядачі в проекті також присутні контролери для повертання даних в форматі JSON або XML які використовуються для витягування даних про точки заправних станцій, додаткової інформації про них та побудови маршруту.

В проекті для цього використовується контролер *WarehouseApiController*, який наслідується від *BaseApiController* в якому наявні методи для отримання точок та інформації про них.

При виборі даних по параметру, наприклад по унікальному ідентифікатору запису, ми передаємо цей унікальний ідентифікатор в метод, де

дані фільтруються та знаходиться бажаний елемент. Методи додавання та редагування для виконання своєї логіки приймають модель об’єкта, з яким потрібно виконати операцію.

3.1.4 Реалізація клієнтської частини

Коли користувач заходить на сайт, де розміщена система, перш за все він бачить перед собою вікно авторизації. Без авторизації користуватися системою неможливо, оскільки це порушує конфіденційність та з міркувань безпеки краще проводити облік тих, хто заходить в систему. Після входу працівник може бачити меню з посиланнями на різний функціонал системи. На даному етапі розробки доступні такі функції як:

- розміщення нових предметів;
- пошук товару на складі;
- перегляд доступних локацій;
- відвантаження товару;
- адміністрування.

В залежності від ролей ці кількість цих функцій може варіюватись. Для адміністратора додатку буде доступний весь функціонал, тоді як для звичайного працівника тільки перші три пункти. Окрім того, функціонал на самих сторінках може відрізнятись.

Розглянемо детальніше:

Сторінка розміщення товарів. На цій сторінці вводяться параметри товару, який щойно прибув на склад. Після введення усіх габаритів вантажу та натискання кнопки «Find Location» система шукає найбільш правильне місце для нього. На поточній стадії виводиться буквенне позначення локації.

Пошук товарів на складі. Для того, щоб відвантажити товар, або навіть просто перевірити його, потрібно спочатку знайти дану одиницю на складі. В умовах великої площі така задача стає досить складною. Однак, після розміщення, система записує всі дані про товар в базу і, знаючи певний

параметр (UPC, код, або внутрішній UPC) його можна знайти без проблем та глянути повну інформацію разом з локацією, в якій він розташований.

На сторінці пошуку виводиться список усіх товарів, які є на даний час на складі. Зверху знаходиться форма пошуку: текстове поле та випадаючий список, в якому можна вибрати поле за яким потрібно шукати.

Перегляд доступних локацій. На цій сторінці виводиться список ще не зайнятих товаром локацій. Можна також відфільтрувати локації за розміром. Для цього потрібно вибрати з випадаючого списку потрібний розмір (підтягуються з таблиці «*Size*»).

Відвантаження товару. Перед тим, як товар покине територію складу, його необхідно відвантажити. Для цього створена окрема сторінка, на якій потрібно знайти товар, а потім оформити для відвантаження. Після завершення процесу, в базі даних статус вантажу міняється на «Відвантажений».

Адміністрування. Залежно від ролей (керівник складу або адміністратор) користувач потрапляє на сторінку, де може мати змогу редагувати локації, додавати за необхідності нові та загалом повністю керувати даними системи.

3.1.5 Реалізація підтримки додатку та безперервної інтеграції

Кожен додаток, що розробляється, повинен мати підтримку з боку розробника. Для цього вихідний код додатку має бути розміщений в репозиторії, який в свою чергу повинен бути хмарним, щоб дані зберігалися на сервері і були доступні розробникам звідусіль. Окрім того, бажано налаштувати безперервну інтеграцію додатку (Continuous Integration). Безперервна інтеграція (CI) – це практика розробки, згідно з якою передбачається, що кожне збереження змін модулів або частин додатку супроводжується автоматичною побудовою рішення та розгортанням готової системи на тестових серверах. Це дозволяє командам рано виявляти проблеми.

В системі, що розробляється в даному дипломному проекті підключений git репозиторій. Синхронізація даних відбувається за допомогою хмарного сервісу збереження репозиторіїв Atlassian Bitbucket. Він є безкоштовним та

дозволяє створювати приватні репозиторії, що не будуть доступні іншим користувачам, що не входять в створену групу.

Окрім того, додаток може розміщуватись на IIS хостингу. Розгортання рішення на сервер проходить по FTP протоколу, за допомогою Visual Studio Web One-Click-Publish. Ця технологія дозволяє будувати рішення та копіювати готові файли на IIS сервер в один клік. Для цього потрібно лише змінити налаштування та вказати логін, пароль і шлях до сервера.

В даному розділі було спроектовано модель бази даних, створено саму базу даних. Також було описано розробку логіки контролерів, в яких міститься код, що зв’язує базу даних та представлення. Окрім того, було описано розробку клієнтської частини програми, представлень, що бачить користувач.

3.2 Алгоритм розміщення товарів

3.2.1 Опис алгоритму розміщення товарів

Отже, для того щоб система автоматично визначала місце, куди необхідно поставити новий товар що прибув на склад, необхідно спочатку визначитися з форматом даних, які будуть обраховуватись.

Основними параметрами товару є його габарити, тобто довжина, ширина, висота, а також маса. Саме вони й будуть формувати вхід нейромережі. Відповідно на основі цих даних буде формуватися так звана категорія, тобто буде шукатися вірний розмір місця для цього товару.

Клас, екземпляр якого подаватиметься на вхід міститиме наступні поля:

– *Length* (довжина);
– *Width* (ширина);
– *Height* (висота);
– *Weight* (маса).

Клас навчання буде схожим до попереднього, але матиме на вході ще одне поле *SizeCode*:

– *Length* (довжина);

– *Width* (ширина);
– *Height* (висота);
– *Weight* (маса);
– *SizeCode* (код розміру).

Після знаходження правильного розміру система далі буде проводити перевірку чи дійсно підходять введені габарити нової одиниці вантажу до знайденого розміру. При успішному проходженні перевірки в базі даних, з таблиці локацій буде шукатися вільна локація з таким розміром. У випадку, якщо локація не знайдена, буде обрано найближчий до знайденого, але трохи більший по об’єму і кожному з параметрів розмір та алгоритм ще раз прорахує вільну локацію.

3.2.2 Навчання і тестування алгоритму розміщення товарів

Для тестування інтелектуальної системи розміщення товарних позицій на складі тестовими даними розмірів, яка налічує 250 можливих варіантів комірок для розміщення товару заповнено базу. Крім того, в БД занесено інформацію про локації тестового складу. На основі цих даних створено тестовий навчальний датасет, формат якого наведений в попередньому пункті. Він складається з 1000 комбінацій габаритів і відповідних розмірів. Саме цей датасет використовується для навчання нейронної мережі. Також, створено аналогічний масив даних але без розмірів для перевірки правильності роботи.

Після прогону тестових навчальних даних, а потім і даних без розмірів, було досягнуто точності класифікації в 93,9%. Крім того, якщо система неправильно підбере розмір та локацію в панелі адміністрування є можливість відредагувати її та вибрати правильну. Ці зміни також внесуться в нейромережу та в подальшому вгадування буде тільки покращуватись.

3.3 Вимоги до технічного забезпечення

При роботі з нейронною мережею загальні вимоги особливого значення не грають, однак вони є принциповими при розміщенні веб системи. Для того щоб сервер IIS справно працював та витримував достатню кількість користувачів потрібна комп’ютерна система повинна мати наступні характеристики:

- процесор – 2.9 Ггц, 4 ядра ЦП або краще;
- оперативна пам’ять не менш ніж 8096 Мб;
- не менше ніж 64 ГБ ПЗУ;
- доступ до мережі Інтернет.

Якщо ж розглядати вимоги до апаратного забезпечення клієнтської частини системи, то вони є мінімальними на даний час:

- процесор – 1.6 Ггц, 2 ядра ЦП або краще;
- оперативна пам’ять не менш ніж 2048 Мб;
- не менше ніж 4 ГБ ПЗУ;
- доступ до мережі Інтернет.

3.4 Вимоги до програмного забезпечення

Щодо програмних вимог, то серверна комп’ютерна система повинна мати наступні програмні рішення:

- базу даних MS SQL 2016;
- .NET Framework версії 4.6 і вище;
- IIS Server;
- операційну систему Windows Server.

Клієнтська частина системи може бути будь-якою. Тобто підходить будь-яка ОС, яка має в собі веб-браузер, або на яку можна його встановити. Єдиним нюансом є те, що браузер повинен підтримувати JavaScript та HTML 5.

3.5 Керівництво користувача

При запуску системи користувачеві перш за все потрібно авторизуватись. Після вводу логіну та пароля та натискання кнопки «Login», користувач потрапляє на головний екран.

На головному екрані містяться посилання на: розміщення нових предметів, пошук товару на складі, перегляд доступних локацій, відвантаження товару, адміністрування системи. В залежності від ролей користувача, цих посилань може бути менше.

На екрані розміщення нових товарів є 7 полів: довжина, ширина, висота, фактична маса, код UPC, назва, опис. Останні 2 не є обов'язковими. Після введення даних потрібно натиснути кнопку «Find Location». Якщо система успішно віднайшла локацію, то вона відобразиться на екрані. У випадку помилкового підбору локації адміністратор або керівник складу може відредагувати цю інформацію.

Сторінка пошуку товару містить список товарів, що знаходяться в даний момент часу на складі, та поля пошуку за певними параметрами. Пошук здійснюється введенням в поле пошукового рядка та вибором з випадаючого списку поля, за яким він буде здійснюватися. Після натискання кнопки «Search» на екрані з'явиться інформація про товар.

Сторінка відвантаження товару змінює його статус. Вона аналогічна пошуку, однак після успішного знаходження доступна опція відвантаження. Статус товару змінюється на «Відвантажений», а сам він перестане відображатися в списку наявних на складі одиниць.

На сторінках адміністрування доступний функціонал повної зміни локацій, розмірів, редагування складських одиниць. Крім того, адміністратор має змогу додавати нових користувачів до системи, видавати їм необхідні ролі та редагувати інформацію. Також інформацію про статуси товарів може змінювати як адміністратор системи, так і керівник складу.

Висновки по розділу

В третьому розділі описано етап розробки алгоритмічного та програмного забезпечення системи інтелектуального розміщення товарних позицій на складі. В результаті проведеної роботи описано і створено базу даних, модель БД та основні сутності. Відповідно до обраної архітектури системи розроблено серверну частину на ASP.NET MVC та візуальну частину, з якою буде працювати користувач. Окрім того, спроектовано і реалізовано нейронну мережу для оптимального та ефективного пошуку місця для нового товару на складі. Наведено технічні і програмні вимоги до обох частин системи: серверної та клієнтської. Складено інструкції, в яких покроково вказано, як працювати з системою, для різних груп користувачів, в тому числі й для адміністраторів додатку.

РОЗДІЛ 4. МАРКЕТИНГОВИЙ АНАЛІЗ СТАРТАП-ПРОЕКТУ

4.1 Опис ідеї проекту

Ідея проекту полягає в створенні системи керування складським приміщенням, що буде містити інтелектуальну систему, за допомогою якої можна в автоматичному режимі знаходити місце для нових товарів на складі. Розглянемо зміст ідеї, можливі напрямки застосування, основні переваги, які зможе отримати користувач представлено у табл. 4.1.

Таблиця 4.1. Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користучача
Робоча система управління складським приміщенням завдяки якій можна в автоматичному режимі розподіляти новий товар	Малий та середній бізнес	Користувачі можуть керувати товарами та вантажами, які перебувають на складі. Окрім того, місце для нового товару вибирається автоматично та точно, що дозволить ефективніше використовувати корисне місце
	Логістика	

Даний проект відрізняється тим, що спеціалізується на зручності системи керування складом та мінімізації людських операцій. Далі проведено аналіз потенційних техніко-економічних переваг ідеї (чим відрізняється від існуючих аналогів та замінників). Визначено перелік техніко-економічних властивостей та характеристик ідеї. На ринку наявні декілька конкурентних продуктів, які мають схожий функціонал, однак все-ж таки відрізняються за певними критеріями.

Проведено порівняльний аналіз показників: для власної ідеї визначаються показники, що мають гірші значення (W, слабкі), аналогічні (N, нейтральні) значення, кращі значення (S, сильні) (табл. 4.2)

Таблиця 4.2. Опис ідеї стартап-проекту

№	Техніко-економічні характеристики ідеї	Продукція конкурентів				*W* (слабка сторона)	*N* (нейтральна сторона)	*S* (сильна сторона)
		Мій проект	Logiwa WMS	Cadence	SphereWMS			
1	Швидкість роботи	Швидка	Повільна	Повільна	Середня	Безпосередня робота з базою даних	Простота архітектури	Накопичення інформації в одному місці
2	Зручність використання	Зручно	Зручно	Відсутня	Зручно	Мала к-сть налаштувань	Перспективи розвитку UI	Простота наявного функціоналу
3	Вимоги до системи	Середні	Мінімальні	Високі	Середні	Відсутня оптимізація під старі системи	Оптимізована робота через веб-додаток	Актуальність програми для нових систем через використання новітніх бібліотек
4	Кросплатформенність	Наявна	Наявна	Відсутня	Наявна	Серверна частина не є кросплат-форменною	Можливість розширення API для створення нового функціоналу	Повна кросплатформенність, оскільки системою можна користуватися з веб-браузера
5	Функціонал	Мало функцій	Багато функцій	Мало функцій	Достатньо функцій	Мала кількість функціоналу	Гнучке API і модель	Можливість розширювати функціонал достатньо швидко за потреби

4.2 Технологічний аудит ідеї проекту

Таблиця 4.3. Технологічна здійсненність ідеї проекту

№	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Робота з накопиченим масивом інформації	Робота через API бази даних	База даних та API	Повністю відкрита для роботи з інформацією
2	Робота з інформацією, яку вводить користувач	API для роботи з обробки даних	Розроблені бібліотеки для роботи нейронної мережі	Повністю відкритий програмний код та використання готових бібліотек
3	Робота з тестовим масивом інформації та даними, які вводить користувач	Використання всіх вище технологій	База даних та API. Розроблені бібліотеки для роботи нейронної мережі	Повністю відкрита для роботи з інформацією. Повністю відкритий програмний код та використання готових бібліотек
Обрана технологія реалізації ідеї проекту:3				

Висновок: технологічна реалізація продукту – можлива, вибрана технологія №3, яка може нам допомогти розробити якісний продукт з використанням комбінації технологій, які працюють з великим відкритим масивом даних та обробкою як наявних даних, так і нових даних, введених користувачем. Крім того реалізоване зберігання результатів безпосередньо користувачем системи. Техніко-економічні характеристики через на початковому етапі введення системи в експлуатацію будуть відставати від світових аналогів, які використовуються для іншого призначення та цілей.

4.3 Аналіз ринкових можливостей запуску стартап-проекту

Визначення ринкових можливостей, які можна використати під час ринкового впровадження проекту, та ринкових загроз, які можуть перешкодити

реалізації проекту, дозволяє спланувати напрями розвитку проекту із урахуванням стану ринкового середовища, потреб потенційних клієнтів та пропозицій проектів конкурентів [22]. Спочатку проводиться аналіз попиту: наявність попиту, обсяг, динаміка розвитку ринку (табл. 4.4).

Таблиця 4.4. Попередня характеристика потенційного ринку

№	Показники стану ринку	Характеристика
1	Кількість головних гравців, од	3
2	Загальний обсяг продаж, грн./ум.од.	25000
3	Динаміка ринку	Стагнація національної економіки. Темпи розвитку світової економіки позитивні, але з ознаками зменшення росту.
4	Наявність обмежень для входу	Відсутні. Конкуренти займають свої певні сфери, які висвітлюють функції, які вказані в дисертації стартапу, тільки як додаткові.
5	Специфічні вимоги до стандартизації та сертифікації	Відсутні
6	Середня норма рентабельності в галузі або по ринку, %	69

Висновок: враховуючи кількість головних гравців по ринку, зростаючу динаміку ринку, невелику кількість конкурентів та середню норму рентабельності можна зробити висновок, що на даний момент, ринок для входження стартап продукту є привабливим та надає змогу реалізувати продукцію на світовому ринку, оскільки на національному ринку економічний розвиток не надає необхідного результату для вдалої економічної реалізації проекту.

Таблиця 4.5. Характеристика потенційних клієнтів стартап-проекту

№	Потреба, що формує ринок	Цільова аудиторія	Відмінності у поведінці цільових груп клієнтів	Вимоги споживачів до товару
1	Потреба занесення інформації про нові одиниці товару, які приходять на склад	Працівники складу	Стартап переважно буде виконувати функції внесення нової інформації в базу даних	Зручність у використанні. Швидка робота системи. Спроможність швидко освоїти як користуватися системою. Можливість впроваджувати користувачам свій функціонал у систему для більшого розвитку проекту серед інших цільових аудиторій.
2	Потреба в ефективному використанні корисного місця складу та збільшення місткості	Власники складу	Стартап переважно буде виконувати функції автоматичного визначення правильного місця розташування нового товару на території складу за допомогою інтелектуальної системи та нейромережі	
3	Потреба в швидкому завантаженні та відвантаженні товару зі складських приміщень	Логістичні компанії	Стартап переважно буде виконувати функції пошуку наявного товару на складах та швидкого відвантаження завдяки зручній адміністративній панелі та навігації	

Таблиця 4.6. Фактори загроз

№	Фактор	Зміст загрози	Можлива реакція компанії
1	Конкуренти	Наявність конкурентів котрі надають схожі рішення	Зменшення ціни на поставлену послугу; Розробка унікальних характеристик товару; Надання ліцензій на обслуговування
2	Кошти на розробку та підтримку продукту	Закінчення грошей та недостатнє фінансування	Залучення додаткових інвесторів, мотивація роботи на перспективу; Ітеративна розробка продукту задля покрокового виведення продукту на ринок та отримання відповіді користувачів
3	Вихід аналогу	Вихід аналогу даного товару може призвести до знецінення та безідейності даного товару	Вихід товару на ринок в коротші строки з не повною, але достатньою, функціональністю для зацікавлення усіх цільових аудиторій; Проведення рекламної компанії

Таблиця 4.7. Фактори можливостей

№	Фактор	Зміст можливості	Можлива реакція компанії
1	Новий продукт	Вихід на ринок, Зменшення монополії, Надання нових рішень у сфері	Розробка нової функціональності; Вихід нової продукції на ринок; Надання різноманітних типів ліцензій в залежності від потреб користувача \ замовника.
2	Вихід аналогу	Надати продукт з певними характеристиками та можливостями що відсутні у компаній конкурентів	Аналіз ринку та користувачів задля задоволення їх потреб та надання функціональності у найкоротші строки за ціну, котра є дешевшою ніж у продуктів-замінників.
3	Зворотній зв’язок від користувачів	Можливість отримання необхідної інформації для вдосконалення продукту	Наявність вхідних даних та реакція на них з боку команди розробників задля задоволення потреб та бажань кінцевих користувачів системи кешування даних.
4	Грошова винагорода за рекламу	Можлива комерціалізація продукту на основі реклами задля отримання грошової винагороди для подальшого розвиту	Точкова комерціалізація продукту; Введення реклами; Ведення додаткових коштів у проект задля його подальшого розвитку.

Таблиця 4.8. Ступеневий аналіз конкуренції на ринку

№	Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1	Тип конкуренції: монополістична	Товар від кожної компанії на ринку, являється недосконалим замінником товару, реалізованого іншими фірмами; На ринку є умови для входу та виходу; Ціна корелює між суперниками;	Розробка продукту з характеристиками, які покривають сфери вживання що не покривають інші товари-замінники; Кореляція цін у відповідності до товарів замінників; Різни типи ліцензій.
2	Рівень конкурентної боротьби: світовий	Всі продукти замінники розроблялись інтернаціональними командами з різних куточків світу, продукти не належать до певної держави, а належать команді розробників	Вихід на ринок збуту продукту з клієнто-необхідною функціональністю; Налагодження маркетингу на основних Інтернет ресурсах задля охоплення великої кількості потенційних користувачів; Надання бета-версій продукту.
3	Галузева ознака: внутрішньогалузева	Даний тип продукту може використовуватися тільки у сфері розробки ІТ додатків \ продуктів	Надання зручного, інтуїтивно зрозумілого інтерфейсу; Підтримка всім відомих методів взаємодії з середовищем розробки; Наявність документації та он-лайн підтримки.
4	Конкуренція за видами товарів: товарно-видова	Дана конкуренція – конкуренція між товарами одного виду.	Впровадження функціональності яка відсутня у товарів-замінників; Спрощення інтерфейсів; Надання підтримки.
5	Характер конкурентних переваг: цінова та не цінова	Цінові переваги – точкова комерціалізація; Не цінова – надання функціональності, що відсутня у товарах-замінниках.	Надання платних ліцензій лише на критично важливу функціональність для клієнта з певним строком підтримки, що зазначена у відповідній ліцензії; Впровадження унікальної функціональності.
6	За інтенсивністю: марочна	Наявність унікального знаку що відрізняє даний продукт від продуктів-замінників	Впровадження власної назви та власного знаку.

Таблиця 4.9. Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	Logiwa WMS, SphereWMS	Cadence	Офіційні представники компаній, реселлери	Форуми, точки продажу	Продукція компанії Cadence
Висновки	Прямі конкуренти намагаються сконцентруватися на інших напрямках свої продуктів.	Потенційні конкуренти мають дуже специфічну клієнтуру до якої зазвичай входять корпоративні клієнти. Конкуренція може відбутися тільки продажах на корпоративний рівень клієнтів	Постачальники диктують умови збереження даних, які захищають приватність користувачів. Також постачальники не дають змогу зловживати етичними нормами.	Клієнти можуть диктувати умови на ринку тільки через повідомлення на форумах або в полі відгуків в точках продажу додатку.	Можливість введення стандартизації можливостей для усіх WMS систем

Проаналізувавши можливості роботи на ринку з огляду на конкурентну ситуацію можна зробити висновок: оскільки кожний з існуючих продуктів не впливає у великій мірі на поточну ситуацію на ринку в цілому, кожний з існуючих продуктів має свою специфічну сферу використання та свої позитивні та негативні сторони щодо рішення певних типів задач, то робота та вихід на даний ринок є можливою і реалізованою задачею [23].

Для виходу на ринок продукт повинен мати функціонал що відсутній у продуктів-аналогів, повинен задовольняти потреби користувачів, мати необхідний та достатній функціонал з конфігурування, підтримку зі сторони розробників та можливість розробки спеціального функціоналу за відповідною ліцензією. Окрім того, потрібно забезпечити користувача адекватної технічною підтримкою та спроможністю розробникам отримувати відгуки від користувачів.

Таблиця 4.10. Обґрунтування факторів конкурентоспроможності

№	Фактор конкурентоспроможності	Обґрунтування
1	Прагматичність	Через запуск стартапу система буде не дуже складно з точки зору архітектури перший час. Через певний період із додаванням функціоналу та оптимізації алгоритмів роботи програмний код буде все складнішим. Такий етап наступить не раніше одного року постійної роботи над проектом.
2	Зручність	Оскільки стартап розробляється на багатьох платформах з різною шириною екранів, то зручність використання системи на різних пристроях буде відігравати не малу роль у спроможності конкурувати з іншими гравцями ринку
3	Швидкість роботи	Швидкість роботи відіграє велику роль для користувачів, оскільки вони не будуть готові чекати декілька хвилин на виведення результату роботи додатку.
4	Оптимізація	Якщо додаток буде дуже часто видавати помилки при роботі, то користувачі не будуть вважати додаток надійним.
5	Налаштування під користувача	Різні люди мають різні звички, які вони використовують, наприклад, якщо є люди, які люблять працювати за додатком де є темні кольори, а є такі люди, які люблять світлі кольори. Можливість редагувати зовнішній вигляд додатку надає значну перевагу серед конкурентів.
6	Відкритість вихідного коду	При наявності вихідного коду будь-який продукт має перспективи розвиватися у багатьох напрямках, особливо таких, які можуть бути неочевидні на перший погляд.
7	Приватність	В останні роки приватність людей та інформація щодо них все частіше зловживається шахраями або великими корпораціями, які потребують погодження з умовами доступу до приватної інформації та її обробки.

Закінчення таблиці 4.10

8	Технічна підтримка	Якщо технічна підтримка компанії буде працювати своєчасно та швидко, то це допоможе зберегти репутацію компанії на відміну від конкурентів, де їй не приділяють увагу.
9	Документація	Будь-який додаток, особливо якщо він має новий функціонал, повинен бути добре роз'яснений своїм користувачам та як його можна використовувати, щоб не мати проблем при подальшій роботі з ним.
10	Ціна	Чим дешевше товар, тим більше шансів що його можуть купити, особливо якщо він працює краще ніж конкурентний товар.

За визначеними факторами конкурентоспроможності (табл. 4.10) проводиться аналіз сильних та слабких сторін стартап-проекту (табл. 4.11). (С.П. – стартап проект, К.1 – Logiwa, К.2 – Cadence, К.3 – SphereWMS)

Таблиця 4.11. Порівняльний аналіз сильних та слабких сторін системи кешування мало змінних даних

№	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з запропонованим						
			-3	-2	-1	0	+1	+2	+3
1	Прагматичність			С.П.	К.2	К.3	К.1		
2	Зручність					К.2	К.1	К.3	С.П.
3	Швидкість роботи					К.1	К.2	К.3	С.П.
4	Оптимізація			К.2		К.1	К.3		С.П.
5	Налаштування під користувача				К.2	С.П	К.3	К.1	
6	Відкритість вихідного коду		К.2	К.3	С.П.	К.1			
7	Приватність				К.3	К.2	К.1	С.П	
8	Технічна підтримка			К.2	К.1	К.3			С.П
9	Документація				К.2	К.3		К.1	С.П
10	Ціна			К.3	К.2	К.1		С.П.	

Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT-аналізу (матриці аналізу сильних (Strength) та слабких (Weak) сторін, загроз (Troubles) та можливостей (Opportunities) (табл. 4.12) на основі виділених ринкових загроз та можливостей, та сильних і слабких сторін (табл. 4.10).

Таблиця 4.12. SWOT аналіз стартап-проекту

Сильні сторони (S): – Прагматичність системи через її легкість роботи; – простота у використанні; – наявність можливості розширення функціоналу; – збереження приватності інформації користувача; – швидкість роботи системи; – зручність	Слабкі сторони (W): – Неоптимізованість алгоритму; – відкритість вихідного коду системи.
Можливості (O): – Зворотній зв’язок з клієнтською базою компанії для спроможності розвивати проект в інші напрямки; – можливість створювати нові модулі системи для розширення функціоналу; – можливість зручної інтеграції цих модулів.	Загрози (T): – Можлива автономна робота алгоритмів; – складність роботи алгоритму при невиявлених випадках використання додатку.

На основі SWOT-аналізу розробляються альтернативи ринкової поведінки (перелік заходів) для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок (табл. 4.9, аналіз потенційних конкурентів). Визначені альтернативи аналізуються з точки зору строків та ймовірності отримання ресурсів (табл. 4.13).

Таблиця 4.13. Альтернативи ринкового впровадження стартап-проекту

№	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Безкоштовне надання певного функціоналу у користування споживачам на обмежений термін	Головний ресурс – люди, даний ресурс - наявний	4-6 місяців
2	Реклама	Залучення власних коштів для реклами товару	2-3 місяці
3	Написання статей та опис товару на відомих ресурсах	Головний ресурс – час, даний ресурс - наявний	2-3 тижні
4	Презентація товару на хакатонах й інших ІТ заходах, на тематичних форумах, виставках по виробничій лінії	Ресурс – час та гроші для участі, наявні	3-6 місяці

Основною альтернативою є презентація товару на хакатонах й інших ІТ заходах, на тематичних форумах, виставках по виробничій лінії.

4.4 Розроблення ринкової стратегії проекту

Таблиця 4.14. Вибір цільових груп потенційних споживачів

№	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Працівники складів, а також їх керівники, які щоденно стикаються з проблемами, які допоможе вирішити розроблювана система.	Присутня	Середній	Низька	Середня

Закінчення таблиці 4.14

2	Власники складів, які стикаються з проблемами ефективності використання корисного місця та збільшення місткості вже побудованих складських приміщень	Присутня	Високий	Низька	Легка
3	Логістичні компанії, які щодня перевозять, доставляють та відвантажують товари	Відсутня	Середній	Присутня	Важка
Які цільові групи обрано: 1, 2					

Відповідно до проведеного аналізу можна зробити висновок, що підходящою цільовою групою для розповсюдження даного програмного продукту є працівники складів, які кожного дня будуть використовувати ключовий функціонал системи. Також, зацікавленими в продукті можуть бути власники складів в цілому та будь-які підприємства котрі використовують схожі програмні продукти. Відповідно до стратегії охоплення ринку збуту товару обрано стратегію масового маркетингу, оскільки для цільових груп в цілому надається стандартизований продукт з можливістю розширення функціональності за домовленістю (відповідно до ліцензії). Для роботи в обраних сегментах ринку необхідно сформувати базову стратегію розвитку (табл. 4.15).

Таблиця 4.15. Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
Надання функціональності що відсутня у товарів-замінників, підтримка клієнтів	Проведення реклами, освітлення унікальної функціональності через інтернет ресурси та інші канали, контакт напряму з споживачами; формування лояльності і прихильності споживачів	Зниження ступеню замінності товару; Прихильність клієнтів; Відмітні властивості товару; Відмітні характеристики товару;	Стратегія диференціації

Таблиця 4.16. Визначення базової стратегії конкурентної поведінки

Чи є проект «першопрохідцем» на ринку	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, які?	Стратегія конкурентної поведінки
Ні, оскільки є товари-заміннники, але дані товари замінники не мають деякого необхідного функціоналу	Так, ціль компанії знайти нових споживачів та, частково, забрати існуючих у конкурентів задля задоволення потреб останніх	Компанія частково копіює характеристики товару конкурента, основна ціль компанії розробка нового унікального функціоналу, з підтримкою основного функціоналу конкурентів	Стратегія заняття конкурентної ніші

Таблиця 4.17. Визначення стратегії позиціонування

№	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту
1	Зручність	Диференціація	Спроможність швидше проводити складські процеси	Спроможність економити час на пошук нової локації
2	Алгоритми роботи	Заняття конкурентної ніші	Перспектива розвитку проекту	Розвиток в науці
3	Документація	Диференціація	Зручність користування системою через відкриті навчальні статті та документацію	Зручність у користуванні

Відповідно до проведеного аналізу можна зробити висновок, що стартап-компанія вибирає як базову стратегію розвитку – стратегію диференціації, як базову стратегію конкурентної поведінки – стратегію заняття конкурентної ніші.

4.5 Розроблення маркетингової програми стартап-проекту

Таблиця 4.18. Визначення ключових переваг концепції потенційного товару

№	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
1	Приймання нового товару на склад	Швидкість внесення нового товару	Зручність та швидкість роботи
2	Знаходження місця для нового товару на складі	Автоматичне виконання даної операції точніше за людину	Повністю інтелектуальне та автоматичне виконання пошуку локації для нового товару на складі
3	Відвантаження товару зі складу	Швидке відвантаження товару зі складу	Швидкий пошук та знаходження товару, формування документів та відвантаження товару в одне натискання кнопки

Таблиця 4.19. Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>1. Товар за задумом</td><td colspan="3">Система інтелектуального пошуку місця для нових товарів на складі</td></tr>
<tr><td rowspan="10">2. Товар у реальному виконанні</td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх/Тл/Е/Ор</td></tr>
<tr><td>Зручність</td><td>Нм</td><td>Е</td></tr>
<tr><td>Швидкість роботи</td><td>Нм</td><td>Тх</td></tr>
<tr><td>Оптимізація</td><td>Нм</td><td>Тх</td></tr>
<tr><td>Ціна</td><td>Нм</td><td>Е</td></tr>
<tr><td>Документація</td><td>Нм</td><td>Тл</td></tr>
<tr><td>Технічна підтримка</td><td>Нм</td><td>Тх</td></tr>
<tr><td>Приватність</td><td>Нм</td><td>Тз</td></tr>
<tr><td>Налаштування під користувача</td><td>Нм</td><td>Ор</td></tr>
<tr><td>Функціональність</td><td>Нм</td><td>Е</td></tr>
<tr><td rowspan="2">3. Товар із підкріпленням</td><td colspan="3">До продажу: наявна повна документація, акції на придбання декількох ліцензій, знижки для певних сегментів на покупку товару</td></tr>
<tr><td colspan="3">Після продажу: додаткова підтримка спеціалістів налаштування, підтримка з боку розробника</td></tr>
<tr><td colspan="4">За рахунок чого потенційний товар буде захищено від копіювання: захист інтелектуальної власності, патент</td></tr>
</table>

Наступним кроком є визначення цінових меж, якими необхідно керуватись при встановленні ціни на потенційний товар (остаточне визначення ціни відбувається під час фінансово-економічного аналізу проекту), яке передбачає аналіз ціни на товари-аналоги або товари субститути, а також аналіз рівня доходів цільової групи споживачів (табл. 4.20). Аналіз проводиться експертним методом.

Таблиця 4.20. Визначення меж встановлення ціни

Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
50 тис. грн	75 тис. грн	280 тис. грн	25-70 тис. грн

Наступним кроком є визначення оптимальної системи збуту, в межах якого приймається рішення (табл. 4.21):

– проводити збут власними силами або залучати сторонніх посередників (власна або залучена система збуту);
– вибір та обґрунтування оптимальної глибини каналу збуту;
– вибір та обґрунтування виду посередників.

Таблиця 4.21. Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
Вибір послуг на сайті, оплата, постачання послуг	-	Виробник-споживач	Web-сайт

Останньою складової маркетингової програми є розроблення концепції маркетингових комунікацій, що спирається на попередньо обрану основу для позиціонування, визначену специфіку поведінки клієнтів (табл. 4.22).

Таблиця 4.22. Концепція маркетингових комунікацій

Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
Знають, які саме послуги треба для вирішення задач	Веб-сайт, телефон, зустрічі	Підтримка клієнтів, індивідуальний підхід	Донесення переваг до клієнтів	Допомога у виборі бібліотеки машинного навчання

Як результат було створено ринкову (маркетингову) програму, що включає в себе визначення ключових переваг концепції потенційного товару, опис моделі товару, визначення меж встановлення ціни, формування системи збуту та концепцію маркетингових комунікацій.

Висновки по розділу

В четвертому розділі описано стратегії та підходи з розроблення стартап-проекту, визначено наявність попиту, динаміку та рентабельність роботи ринку, як висновок було вказано що існує можливість ринкової комерціалізації проекту. Розглянувши потенційні групи клієнтів, бар’єри входження, стан конкуренції та конкурентоспроможність проекту було встановлено що проект є перспективним. Розглянуто та вибрано альтернативу впровадження стартап-проекту та доведено доцільність подальшої імплементації проекту.

ВИСНОВКИ

1. У ході виконання магістерської дисертації було розглянуто питання пов’язані з необхідністю та актуальністю створення системи інтелектуального розміщення товарних позицій на складі на основі нейронних мереж та технології машинного навчання. Наведено характеристику предметного середовища та обґрунтовано причину розробки системи. Описано вимоги до додатку даного виду та технології створення системи.

На основі даних, отриманих в процесі аналізу, сформульовано задачу створення системи розумного розміщення товарних позицій на складі на основі нейронних мереж та технології машинного навчання.

2. Для розробки інтелектуальної системи було використано трирівневу архітектуру додатку з трьома основними частинами: базою даних, серверною та клієнтською. Як мова програмування використовується C#. Серверна частина реалізована за допомогою технології ASP.NET з використанням шаблону проектування MVC. Розміщується додаток на сервері IIS, що робить доступним його з будь-якого пристрою, який має доступ до мережі Інтернет. Для написання частини представлень було використано розмітку HTML, таблиці каскадних стилів CSS, мову програмування Javascript та фреймворки візуальної частини Bootstrap. Розробка велася в середовищі програмування IDE Microsoft Visual Studio 2019. Всі наведені вище технології є абсолютно безкоштовними і широко вживаними. Це дозволяє при необхідності легко розшити функціонал системи в разі виникнення потреби.

3. Розроблена модель бази даних, яка дає змогу протестувати основний та найбільш критичний функціонал системи інтелектуального розміщення товарних позицій на складі. Для управління базою даних було обрано MS SQL Server 2016. З програмного коду серверного рівня доступ до бази даних відбувався за допомогою фреймворка Entity Framework версії 6.

4. Окрім того, реалізовано модуль нейронної мережі. Завдяки йому система може в автоматичному режимі правильно визначати локацію для

оптимального та найбільш ефективного розміщення нового товару на території складу. Після навчання та проведеного тестування точність визначення склала близько 93%.

5. Розроблена та наведена інструкція користувача по експлуатації системи. Наведено приклади сторінок та використання створеного додатку, складено покрокову інструкцію. Розглянуто основний функціонал та способи його використання.

6. Проведений маркетинговий аналіз стартап-проекту. Проведено опис ідеї проекту, технологічний аудит ідеї проекту, аналіз ринкових можливостей запуску стартап-проекту. Розроблено ринкову стратегію проекту та маркетингову програму стартап-проекту.

Результати роботи над магістерською дисертацією опубліковані у статті.

Створена система має полегшити щоденну роботу працівника складу, а також дозволить більш ефективно використовувати площу складських приміщень, що, у свою чергу, дасть можливість здешевити вартість користування складом. Відповідно, це тільки позитивно буде відображатися на ціні кінцевого продукту.

ПЕРЕЛІК ПОСИЛАНЬ

1. Warehouse Management: A Complete Guide to Improving Efficiency and Minimizing Costs in the Modern Warehouse: science book / Gwynne Richards — Kogan Page Publishers, 2011 — 344 p.
2. Warehouse Management: Automation and Organisation of Warehouse and Order Picking Systems: book / Michael Hompel, Thorsten Schmidt — Springer Science & Business Media, 2006 — 356 p.
3. A Supply Chain Logistics Program for Warehouse Management: book / David E. Mulcahy, Joachim Sydow — CRC Press, 2008 — 552 p.
4. Supply Chain Management Best Practices: book / David Blanchard — John Wiley & Sons, 2010 — 320 p.
5. Artificial Neural Networks: Learning Algorithms, Performance Evaluation, and Applications: science book / Nicolaos Karayiannis, Anastasios N. Venetsanopoulos — Springer Science & Business Media, 2013 — 440 p.
6. Artificial neural networks: science book / B Yegnanarayana — New Delhi : Prentice-Hall of India, 1999 — 461 p.
7. Neural Network Modeling: Statistical Mechanics and Cybernetic Perspectives / P. S. Neelakanta, Dolores DeGroff — CRC Press, 1994 — 256 p.
8. Neural Networks: An Introduction / Berndt Müller, Joachim Reinhardt, Michael T. Strickland — Springer Science & Business Media, 1995 — 331 p.
9. Neural Network Computing for the Electric Power Industry: article / Dejan J. Sobajic — Psychology Press, 2013 — 240 p.
10. Говоровський С.В., Базалій М.Ю. Інтелектуальна система розміщення товарних позицій на складі Вчені записки Таврійського Національного Університету Імені В.І. Вернадського. Том 30 (69) №6 2019. С. 56–60.
11. Automatic Generation of Neural Network Architecture Using Evolutionary Computation: science book / E. Vonk, L. C. Jain, Ray P. Johnson — World Scientific, 1997 — 182 p.

12. Adaptive Neural Network Control of Robotic Manipulators, book / Tong Heng Lee, Christopher John Harris — World Scientific, 1998 — 381 p.
13. Machine Learning: An Artificial Intelligence Approach: science book / Ryszard S. Michalski, Jaime G. Carbonell, Tom M. Mitchell — Elsevier, 2014 — 572 p.
14. Machine Learning: An Algorithmic Perspective: book / Stephen Marsland — CRC Press, 2011 — 406 p.
15. Microsoft SQL Server 2016 Reporting Services: science book / Brian Larson — 5th edition — McGraw Hill Professional, 2016 — 793 p.
16. Professional Microsoft SQL Server 2016 Reporting Services and Mobile Reports: book / Paul Turley — John Wiley & Sons, 2017 — 816 p.
17. C# 8.0 and .NET Core 3.0 – Modern Cross-Platform Development: book / Mark J. Price — Packt Publishing Ltd, 2019 — 818 p.
18. Professional ASP.NET MVC 5: science book / Jon Galloway, Brad Wilson, K. Scott Allen, David Matson — John Wiley & Sons, 2014 — 624 p.
19. Hands on with ASP.NET MVC: Covering MVC 6: book / Rahul Sahay — Vij Books India Pvt Ltd, 2014 — 506 p.
20. IIS 8 Administration: The Personal Trainer for IIS 8.0 and IIS 8.5: book / William Stanek — Stanek & Associates, 2015 — 368 p.
21. An Introduction to the Bootstrap: book / Bradley Efron, R.J. Tibshirani — CRC Press, 1994 — 456 p.
22. Managing Startup Enterprises in Emerging Markets: Leadership Dynamics and Marketing Strategies: book / Ananya Rajagopal — Springer Nature, 2019 — 182 p.
23. 101 Startup Lessons: An Entrepreneur's Handbook: book / George Deeb, Red Rocket Ventures — BlogIntoBook.com, 2013 — 150 p.
24. Розроблення стартап-проекту [Електронний ресурс] : Методичні рекомендації до виконання розділу магістерських дисертацій для студентів інженерних спеціальностей / За заг. ред. О.А. Гавриша. – Київ : НТУУ «КПІ», 2016. – 28 с.

ДОДАТКИ

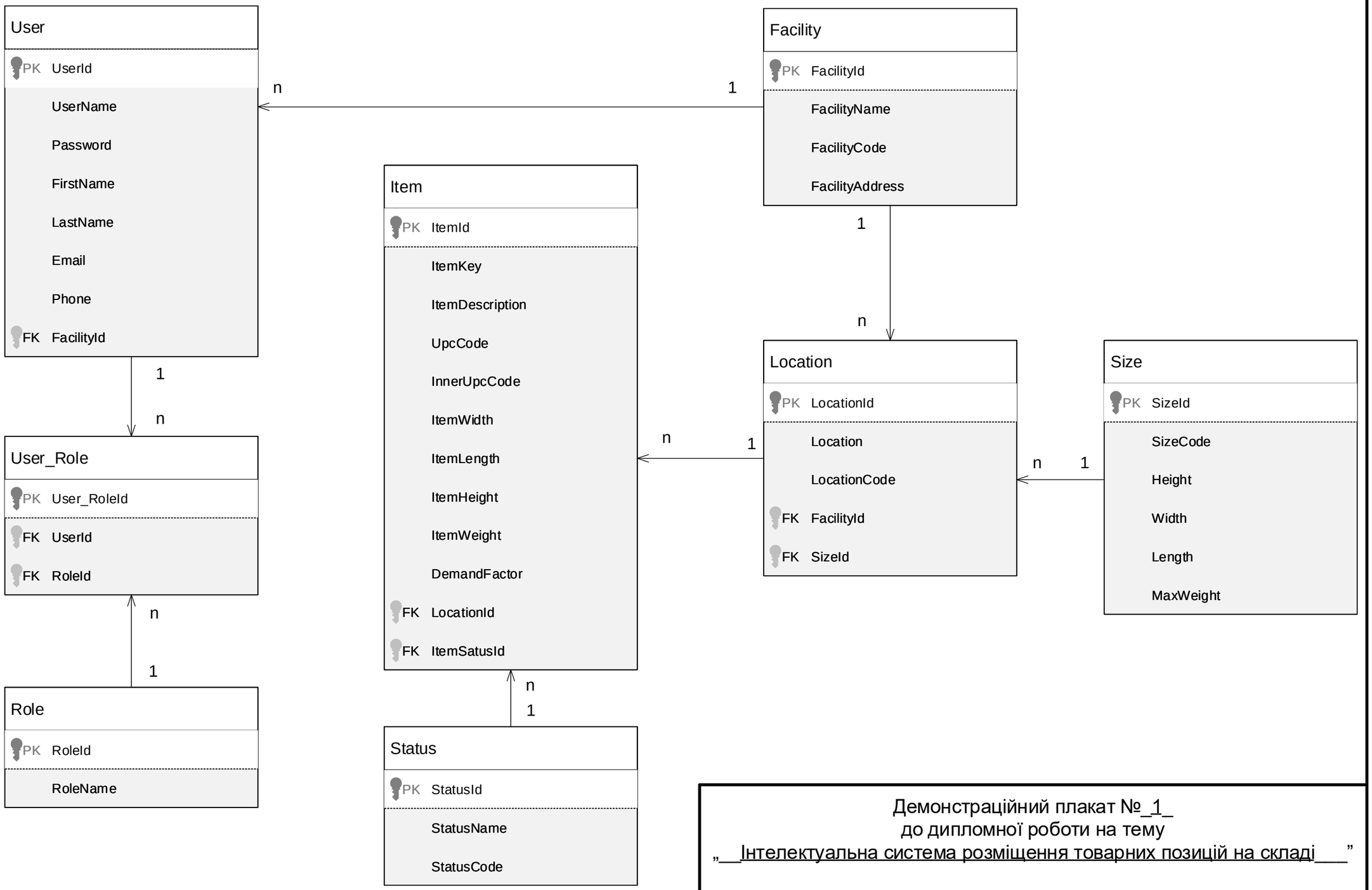
ДОДАТОК А

Результати перевірки на співпадіння

ДОДАТОК Б

Плакати, графічний матеріал

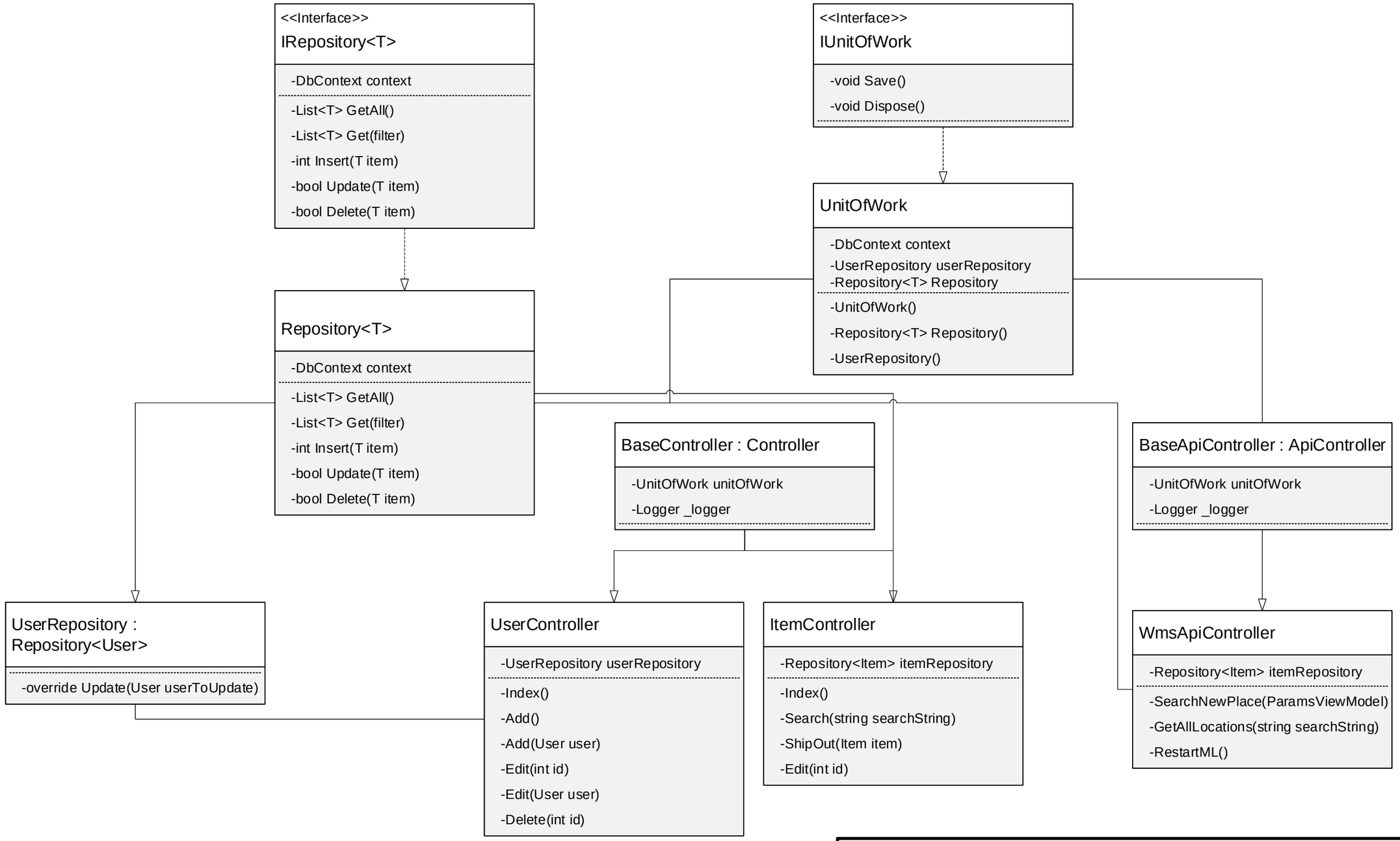
Схема бази даних



Демонстраційний плакат №_1_
до дипломної роботи на тему
„__Інтелектуальна система розміщення товарних позицій на складі___”

Розробив: _Говоровський С.В._
Прийняв: _____

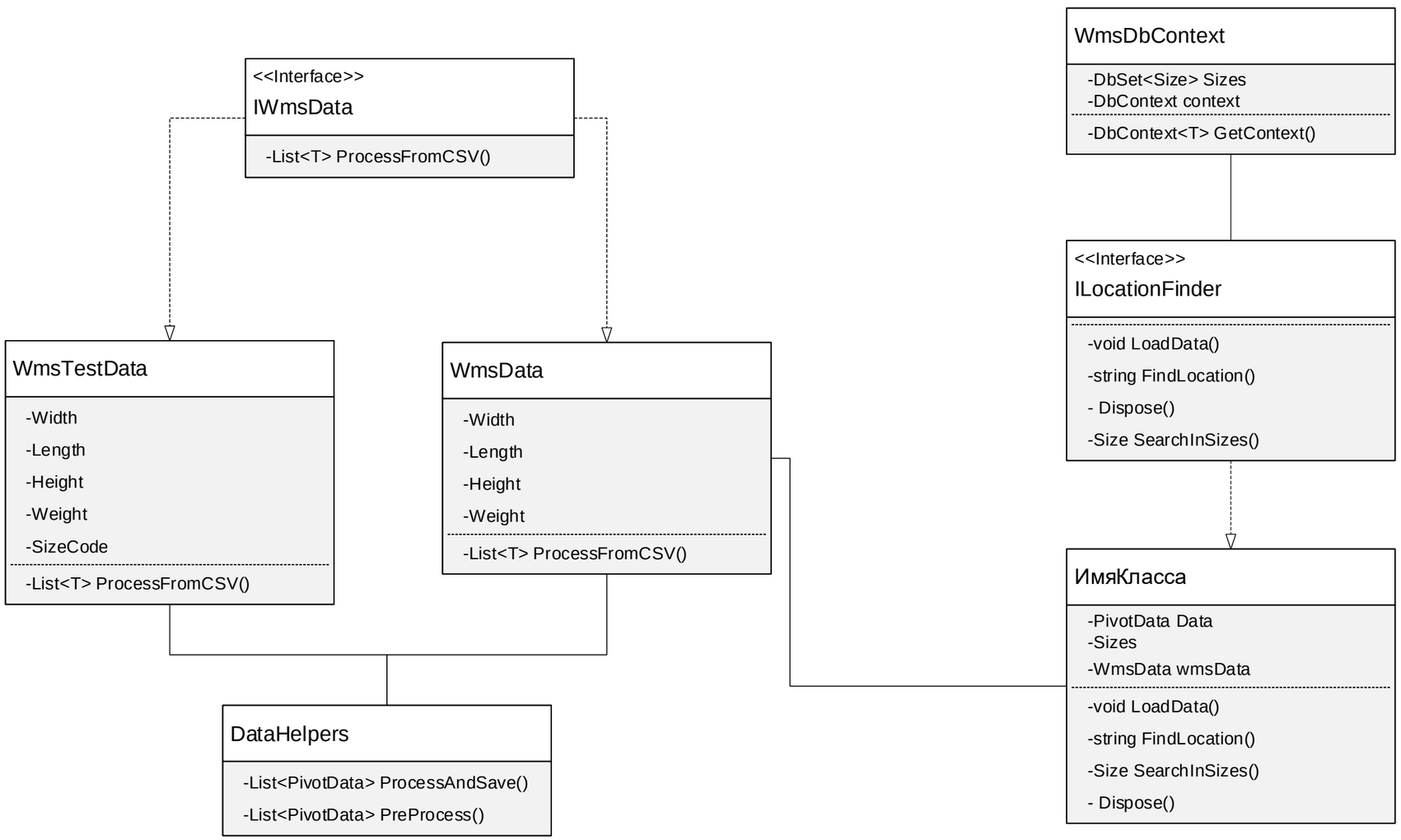
Діаграма класів системи



Демонстраційний плакат №_2_
до дипломної роботи на тему
„__Інтелектуальна система розміщення товарних позицій на складі___”

Розробив: _Говоровський С.В._
Прийняв: _____

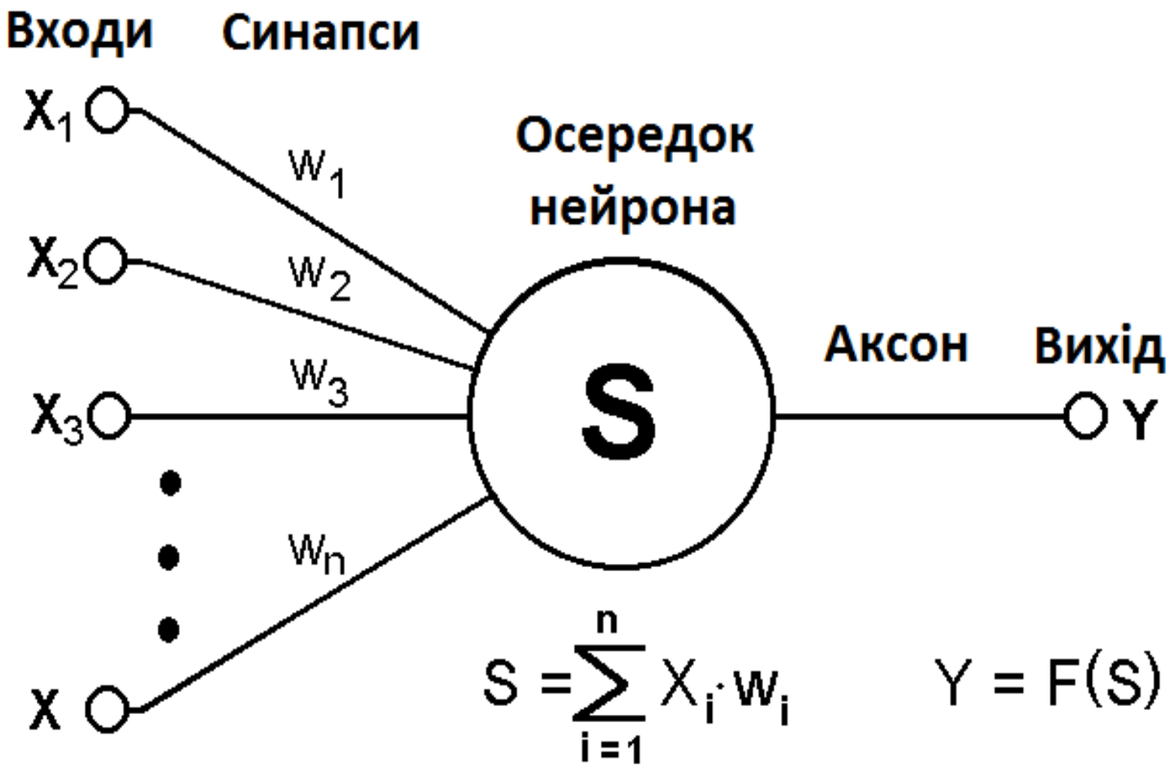
Діаграма класів модуля пошуку місця



Демонстраційний плакат №_3_
до дипломної роботи на тему
„__Інтелектуальна система розміщення товарних позицій на складі___”

Розробив: _Говоровський С.В._
Прийняв: _____

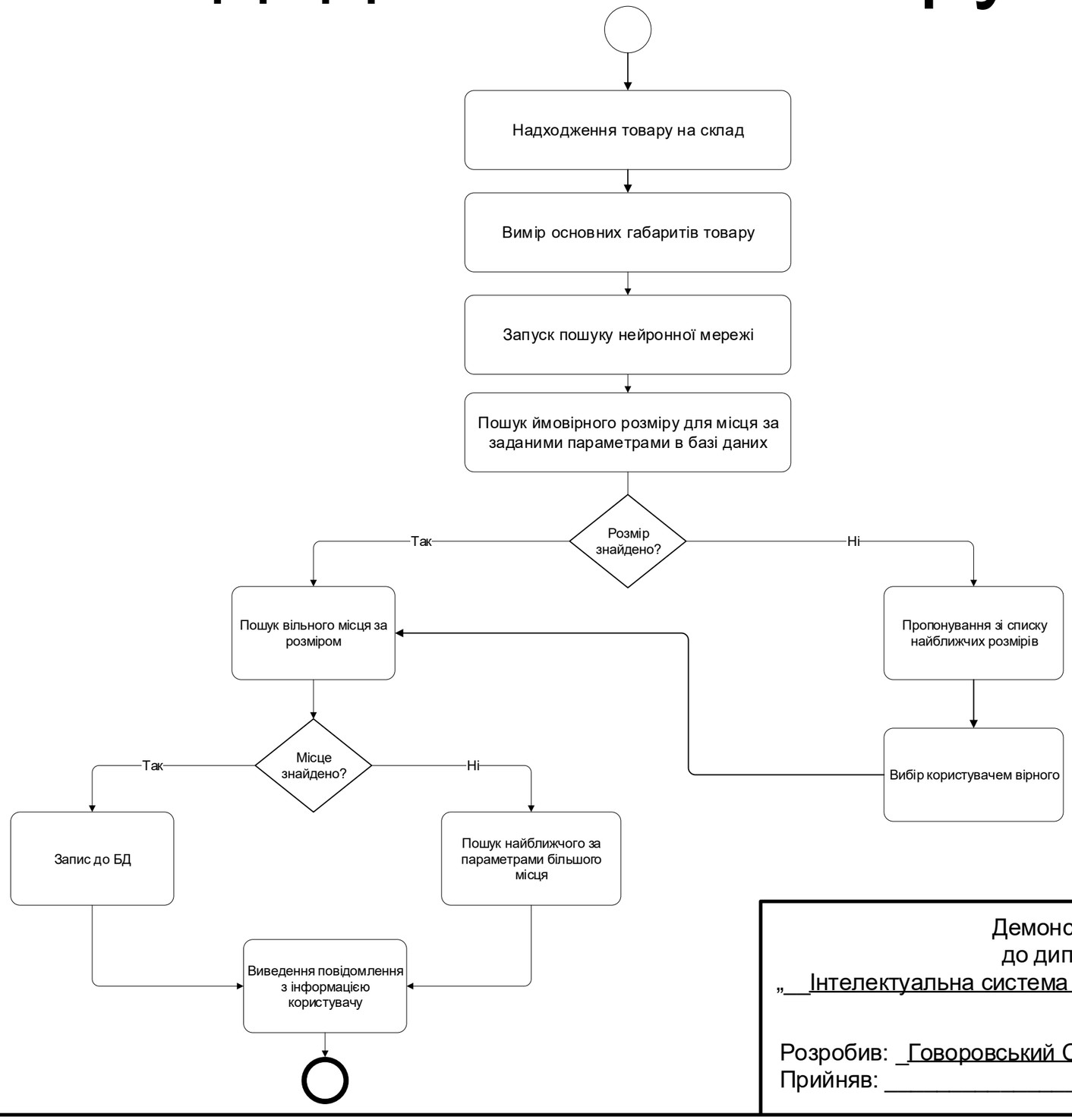
Принцип роботи нейронної мережі



Демонстраційний плакат №_4_
до дипломної роботи на тему
„__Інтелектуальна система розміщення товарних позицій на складі___”

Розробив: _Говоровський С.В._
Прийняв: _____

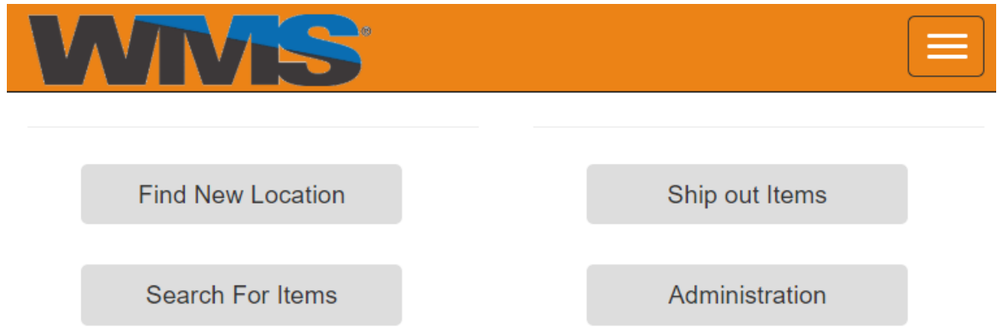
Блок-схема алгоритму додавання товару

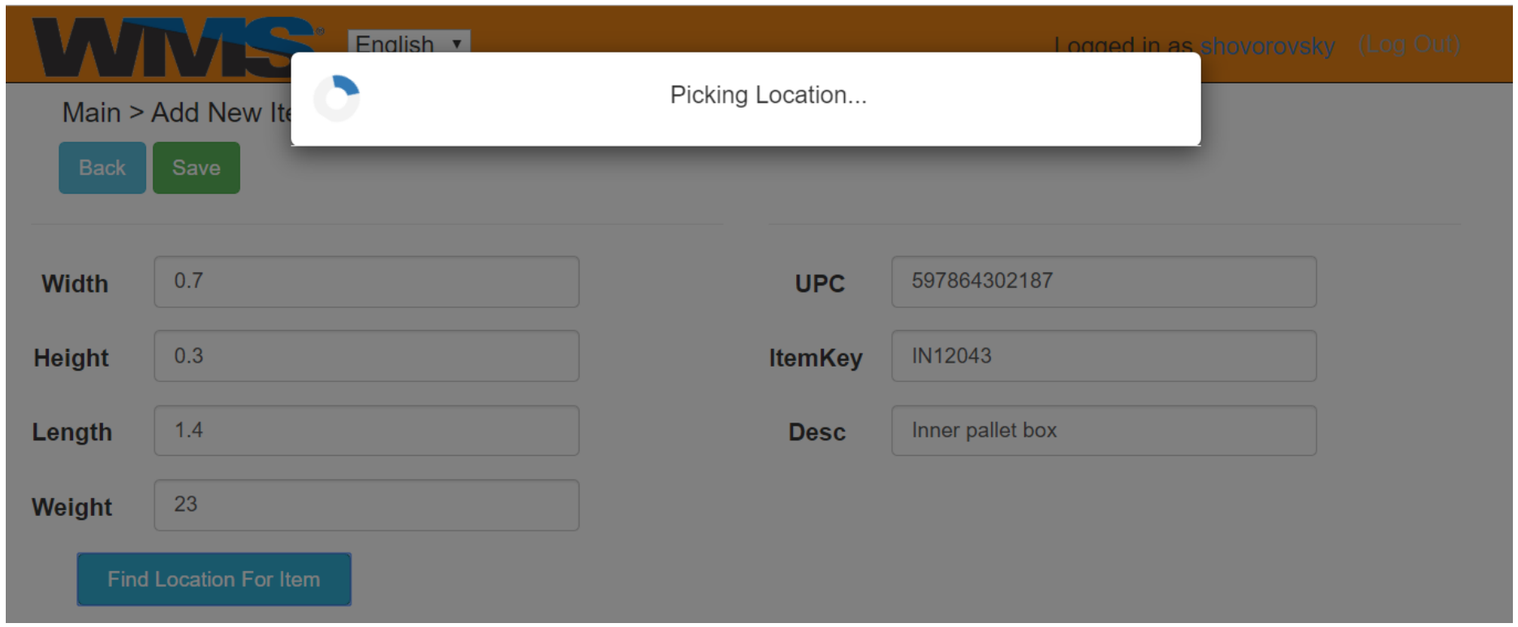


Демонстраційний плакат №_5_
до дипломної роботи на тему
„__Інтелектуальна система розміщення товарних позицій на складі___”

Розробив: _Говоровський С.В._
Прийняв: _____

Інтерфейс користувача системи





Демонстраційний плакат №_6_
до дипломної роботи на тему
„__Інтелектуальна система розміщення товарних позицій на складі___”

Розробив: _Говоровський С.В._
Прийняв: _____